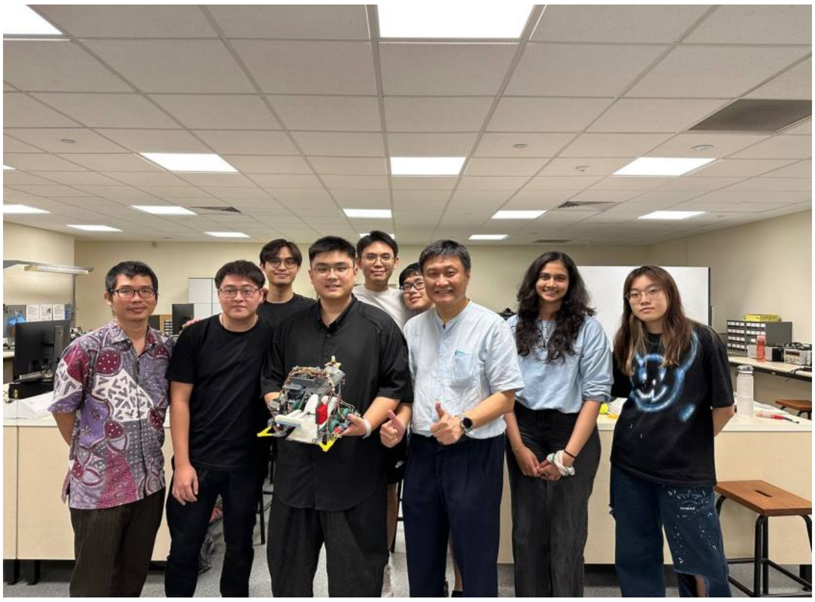
MA4012: **Mechatronics Engineering Design**

Design Project: **Autonomous Tennis Ball Collector**

Tutor: **Dr John Heng**



Team Members:

Liu Qiyuan [Leader]	(U2020275D)
Cao XuBin	(U2021894E)
Daniel Low Teck Fatt	(U2022053E)
Lim Jia Jing	(U2021224D)
Selvam Vinothini Vibhusha	(U2020585D)
Song Ke Yan	(U2020352K)
Wang Xiaoni	(U1923200H)

Abstract

In this era of Industry 4.0, automation has become increasingly significant in the operations of organisations across diverse industrial sectors. The emerging application of automation has enabled companies to streamline manufacturing processes, reduce labour costs, enhance efficiency, and drive market innovations. Among these technologies, autonomous robots have garnered widespread popularity due to their ability to carry loads and navigate independently. This ability of the robots is particularly useful in the logistics sectors.

As an aspiring robotics and mechatronics engineer, it is essential to acquire and hone skills in developing autonomous technologies due to their demand. Thus, for the project in this report, the students are tasked with developing an autonomous vehicle that is capable of searching, collecting and delivering tennis balls to a designated area within a 13-week time frame. Through this, the group of students are able to experience the entirety of the design process from conceptualising mechanical design ideas to prototyping and testing.

This report entails the team's systematic approach to designing an autonomous vehicle that aligns with the project guidelines provided. The report starts off by explaining in great detail the meticulous approach taken to devise a conceptual design idea of the vehicle that adheres to embodiment design rules and principles. The detailed design section outlines the process of transforming the conceptual design idea into an assembled vehicle. Following this, the control system and programming chapters explain the experimental process in finding the optimal sensor positions and efficient search algorithm for swift ball retrieval. To conclude, each student has also provided a reflection detailing their learning experience throughout the project.

Table of Contents

1. Introduction

This chapter provides an overview and outlines the objectives of this project. It also details the vehicle design process, team organisation, project timeline, and includes the list of components provided.

1.1 Overview

The goal of this project is to design and build a compact autonomous vehicle capable of performing the following tasks:

1 Ball searching and retrieval
2 Edge line detection
3 Opponent detection

The autonomous vehicle will then proceed to join a competition where it will compete with vehicles from other teams in an arena (refer to Figure 1) to retrieve 3 tennis balls within a 3-minute time limit. The balls are randomly placed in the arena and the handlers can choose their robots' starting position.

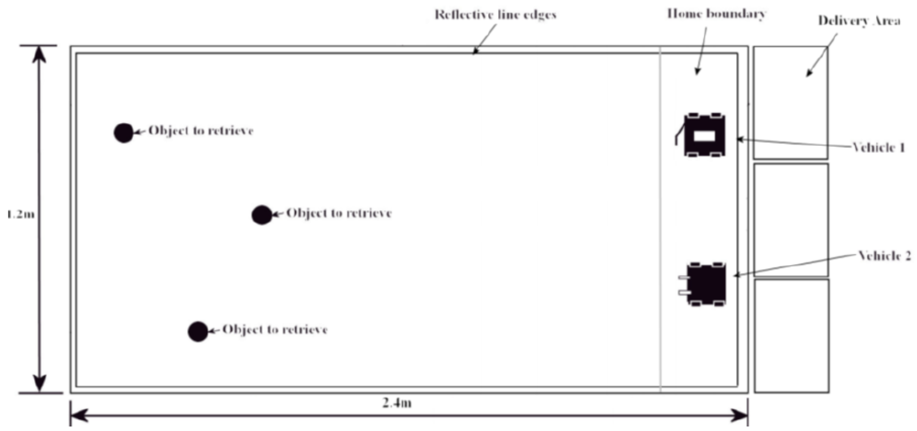


Figure 1: Figure of the arena

The arena features a black-coloured ground, bordered by yellow reflective tape to allow edge detection and enclosed by a 30 mm wall barrier to prevent the robots from moving out of the arena. Each robot can only carry 1 ball at a single time, meaning it must release one if it collects two balls simultaneously. Robots are only allowed to deliver the ball into the delivery box when they are in the home boundary, which is drawn 40 cm away from the delivery area. The delivery area has a boundary wall that requires the ball to be lifted over to be dropped into the delivery box.

1.2 Objectives

The following are the design objectives for the vehicle.

1 The vehicle must be able to locate the ball in the arena
2 The vehicle must be able to retrieve the ball using a collector mechanism
3 The vehicle must be able to carry the ball and deliver it to the ball collection point
4 The vehicle always stays within the boundaries of the arena
5 The vehicle is able to avoid the opponent

1.3 Design Process

We employed a systematic approach to develop the autonomous vehicle. The design process is broken down into 4 distinct steps: Design Definition, Conceptual Design, Embodiment Design, and Detail Design, as illustrated in Figure 2

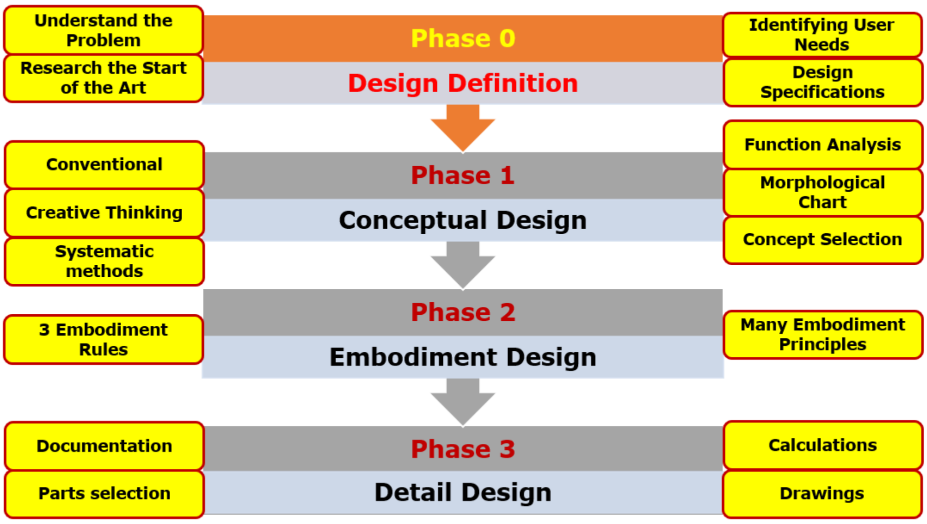


Figure 2: Flowchart of a design process

Defining the design is the first step in the design process, as it helps to identify the stakeholders involved in this competition and establish the design specifications for the robot. It also ensures that the robot meets the provided non-negotiable design requirements. Following that, the conceptual design phase includes the functional analysis of the robot to list the key functions to be carried out and creative thinking of the robot design. The embodiment design phase addresses the principles governing the robot's design, while the detailed design demonstrates the calculations, selection of parts, and detailed drawings of the finalised robot design.

1.4 Team Organisation and Project Timeline

The team consists of 7 members and is divided into two sub-groups: The Mechanical team and the Electrical and Software (E&S) team. The responsibilities of the respective sub-groups are listed in Table 1.

Table 1: Group members and responsibilities of two subgroups

Sub-group	**Mechanical Team**	**Electrical and Software Team**
Group member	Daniel Low Teck Fatt Lim Jia Jing Song Ke Yan	Liu Qiyuan (Leader) Cao Xubin Selvam Vinothini Vibusha Wang Xiaoni
Responsibilities	Vehicle Structure Design Collector Mechanism Design Ramp Design Drive System Design Sensors Installation	Sensor Testing and Calibration Electrical Design Control Algorithm Programming

Table 2 presents a brief timeline of the project.

Table 2: Project Timeline

Week	**Mechanical Team**	**Electrical and Software Team**
2 - 5	Idea Generation and Conceptual Design	
6 - 9	Vehicle structure construction Collector system and ramp design 3D printing of customised parts Drive system installation Sensor’s installations	Sensors and motors testing Sensor’s calibration Electrical design and control algorithm programming
10 - 12	Provide mechanical support to the E&S team during testing Complete CAD assembly of the entire vehicle	Software testing and tuning of software parameters Refinement of control algorithm

1.5 Components Provided

Component	Quantity	Description
VEX ARM Cortex-based Microcontroller	1	A microcontroller built around an ARM-Cortex processor equipped with multiple input and output ports.
VEX USB A-A Tether Cable	1	Connection cable to connect the laptop to the microcontroller for program uploading purposes.
NiMH Battery Pack 7.2V, 3000mAH	1	Rechargeable battery with 7.2V power supply to the microcontroller.
Battery Extension Cable	1	Extends the battery pack cable to reach the microcontroller power port.
VEX Smart Charger and AC Power Cord	1	Plugged into a power socket to charge the NiMH battery pack. The AC power cord converts the power supply to the correct voltage that is compatible with the battery.
VEX Metal and Hardware Kit	1	Contains metal parts to build the autonomous vehicle such as metal plates, bars, angled bars, shafts, gussets, screws, nuts, etc.
VEX Gear Kit	1	Gears in multiple sizes that are used for power transmission from motors.
High Strength Gear Kit	1	Gears in various sizes that are thicker than the VEX Gear Kit and are able to withstand greater forces.
Wheel (Low-friction)	4	Rubber wheel with a radius of 3.5cm
VEX Continuous Rotation Motor (276-2163)	2	Continuous rotation motor controlled by Pulse Width Modulator (PWM) signal generated by the VEX Microcontroller
2-Wire Motor 393	2	Continuous rotation motor controlled by PWM signal from the motor controller using 2-wire cable connection.
Motor Controller 29	2	Sends standard PWM signal to drive the 2-wire motor.
VEX Shaft Encoder	1	Measures the rotational position and rotational motion of the shaft
Sharp Distance Sensor (10 - 80 cm)	3	Measures distance to an object within a range of 10 - 80 cm using an infrared ray.
Sharp Distance Sensor (4 - 30 cm)	1	Measures distance to an object within a range of 4 - 30 cm using an infrared ray.
IR Line Tracking Module	4	Detect the boundary lines by monitoring the change in reflected IR intensity due to variations in surface colour
Limit Switch	4	Sensor that sends an electrical signal to detect the presence of an object when it is pressed.
Digital Compass 1490	1	Sensor that gives the 8 principal directions: 4 cardinal directions (N, E, S, W) and 4 intercardinal directions (NE, NW, SE, SW). This sensor classifies a range of directions as a specific direction rather than identifying a single direction. Thus, it is relatively inaccurate.
Arduino Mega Board	1	An ATmega2560 microcontroller board that comes with 256 KB of flash memory, 8 KB of SRAM, and 4 KB of EEPROM
Reflective Tape on Black Cardboard	1	The reflective tape pasted on the cardboard is in a different colour. It is used to test the IR line tracking module outside of the arena.

2.Design Definition

This chapter explains in detail the identified user needs and non-negotiable design requirements that the vehicle needs to adhere to in order to qualify for the competition.

2.1 Identification of User Needs

The vehicle must be capable of autonomously searching and retrieving tennis balls while simultaneously avoiding the opponent and staying within the arena boundaries. The table below lists the identified user needs that need to be met to achieve the desired functionality of the robot.

Table 4: List of User Needs

Primary Needs	**Breakdown of primary needs**
The vehicle must be able to search for the tennis ball	✓ The vehicle easily activates for prompt search initiation ✓ The vehicle is able to quickly locate the tennis ball ✓ The vehicle is able to accurately locate the position of the tennis ball ✓ There are measures in place to prevent unintentional shutdown
The vehicle must be able to collect the ball	✓ The collector mechanism must be able to retrieve the ball with minimal error ✓ The tennis ball is collected swiftly
The vehicle must be able to deliver the ball to the designated delivery area	✓ The collector mechanism should only release the ball when a control signal is triggered ✓ The collector should be robust enough support the weight of the tennis ball ✓ The collector should be secure enough to prevent the tennis ball from being accidently dropped
The vehicle must always stay within the arena boundaries	✓ The edge detection sensors need to actively detect the yellow edges of the arena ✓ When the sensor is triggered, it must be able to send signals to divert the vehicle from the boundary
The vehicle can avoid opponents	✓ The vehicle needs to detect the opponent ✓ The vehicle needs to be able to move away from the opponent quickly

2.2 Design Specifications

The vehicle is analysed by breaking it down into individual components, each accompanied by specific design requirements. Subsequently, a detailed assessment of the relative importance of each design specification for every component is undertaken. The importance is given a rating of between 1 to 5, with 5 being most important and 1 being least important. The segregated components include the chassis, ball collection mechanism, navigation system and obstacle detection system.

1. Apart from the vehicle design specifications that the team identified, below are the following design requirements that the vehicle must adhere to as given in the design guideline.
2. The autonomous vehicle should be smaller than 30cm x 30cm x 30cm before the start of competition.
3. The vehicle is not allowed to increase its footprint after the start of the competition.
4. The vehicle should comprise more than half of the parts that are provided by the Mechatronics lab.
5. Power supply to the vehicle is limited to a 6-cell AA rechargeable battery pack provided.
6. Sensors are to be deployed at the edges of the vehicle to allow it to detect the boundaries of the arena.
7. The vehicle is to demonstrate good manoeuvrability.
8. Sweeping sharp sensors are to be used to search for the balls.
9. A grasping mechanism should be available to retrieve the ball and allow the vehicle to return to the delivery area to deliver the ball.
10. The vehicle is capable of returning back to the arena to locate more balls.
11. No additional sensors, servos, motors, or batteries are allowed over that of what is provided.
12. No restriction on the number of limit switches used.

2.2.1 Chassis

The chassis serves as the vehicle's backbone, providing a robust foundation for the entire structure. Its structural integrity is crucial in maintaining the vehicle's stability and ensuring safety. Table 5 lists the design specifications and their relative importance.

Table 5: Design Specifications for Chassis

Design Specification	**Importance**
Chassis design must support the weight of collected tennis balls and accommodate additional components without compromising performance	5
Chassis must be able to withstand crashes	4
Chassis must ensure stability for smooth movement and manoeuvrability	5
Chassis must be relatively light	2
Chassis should facilitate easy access to components for maintenance or replacement	3
Chassis design must consider energy-efficiency	3

2.2.2 Ball Collection Mechanism

The ball collection mechanism is the operational core of the autonomous vehicle. The efficiency and reliability of the mechanism is paramount for successful tennis ball retrieval. Its operational integrity is essential not only for effectively collecting tennis balls but also for ensuring the overall success of the robot's mission. The design specifications and their relative importance for the ball collection mechanism are detailed in Table 6.

Table 6: Design Specifications for Ball Collection Mechanism

Design Specification	Importance
The ball collection mechanism must be efficient and precise, ensuring the vehicle can gather tennis balls with minimal errors	5
The ball collection mechanism must be constructed using durable materials	4
The ball collection mechanism must have an accessible design for easy maintenance or replacement	3
The ball collection mechanism carries only 1 ball at a time	5
The ball collection mechanism must be able to swiftly pick up the tennis ball	4
The ball collection mechanism must be able to swiftly drop the tennis ball at the delivery area	3

2.2.1 Navigation System

The navigation system encompasses how the robot searches for the ball, returns the ball to the delivery area as well as how it carries out edge detection. Table 7 lists the design specifications for the navigation system and their relative importance.

Table 7: Design Specifications for Navigation System

Design Specification	Importance
The navigation system must quickly locate tennis balls using sharp sensors	4
The navigation system must accurately locate tennis balls using sharp sensors	5
The robot must be able to return to the delivery area after collecting the tennis ball	5
The robot must be able to quickly return to the delivery area after collecting the tennis ball	3
The navigation system must actively detect arena boundaries to ensure the vehicle stays within the arena boundaries	5
The sharp sensors should be capable of locating the tennis balls even when all sensors on the robot are not functional	1

2.2.1 Obstacle Detection System

The obstacle detection system aids the vehicle in avoiding obstacles. This is especially important for the competition where two robots in the arena will be searching for the same tennis balls. By having an accurate and efficient obstacle detection system, the vehicle can avoid the opponent minimising its risk of collision and damage which can damage its functionality. The design specifications and importance of each is listed in Table 8.

Table 8: Design Specifications for Obstacle Detection System

Design Specification	**Importance**
Ensure accurate obstacle detection to minimise collision risks during competition	4
Implement efficient processing for quick analysis and prompt decision-making	4
Assess collision risk with opponents and prioritise navigation away from potential collisions	5
Seamlessly integrate with the navigation system for coordinated responses to detected obstacles	5

Through this in-depth process of identifying user needs, non-negotiable design requirements and listing out the design specifications for each major functional component of the vehicle, it ensured that all members have a thorough understanding of the design requirements. This also helped to set the stage for brainstorming of vehicle design ideas as the design specifications aided the members with their design thinking process. Moreover, establishing the importance of each design specification made sure members did not lose sight of the big picture by worrying about small details which can be addressed during the optimisation stage of the vehicle design.

3. Conceptual Design

This chapter describes the conceptual phase of the design process, which encompasses functional analysis and creation of a morphological chart. The functional analysis illustrates the complete cycle of tasks to be performed by the robot to successfully retrieve the tennis ball. The morphological chart illustrates different design options for each functional requirement identified in the functional analysis phase. Through a systematic exploration of various design alternatives by considering the pros and cons of each design idea, the morphological chart serves as a valuable tool in generating innovative solutions. The iterative nature of the conceptual phase ensures a thorough examination of design possibilities, laying the groundwork for informed decision-making and refinement in the subsequent stages of the design process.

3.1 Idea Generation

A literature review was first conducted to explore existing design ideas on autonomous vehicles, particularly those that perform object retrieval tasks. Exploring different resources through Google, Youtube and GitHub allowed the team to gauge which ideas were feasible and practical to implement within the given time frame using the components provided. Also, there are several robotic competitions that use the VEX robotics kit, so the team was able to take inspiration on how to assemble the different components.

Subsequently, an intuitive approach was adopted to generate diverse design ideas. Each team member independently brainstormed unique concepts, presenting them to the team for collective evaluation. Through a meticulous assessment of the merits and drawbacks of each design idea, the most promising ideas were kept for further refinement.

This collaborative process ensured that each team member was able to contribute in a meaningful way by bringing forth fresh perspectives and constructive feedback on the ideas presented. This also made certain that all aspects of the vehicle ranging from technical feasibility, practicality and efficiency were considered before finalising the idea. The team ensured to diligently record down all research conducted on a logbook so that important details are not missed and for accountability.

3.2 Functional Analysis

Before vehicle ideas can be generated, it is important to have a basic understanding of the primary task the vehicle has to perform. This includes understanding the flow of materials, energy and control signals involved in carrying out the primary task. The primary task of our vehicle is to collect and deliver the tennis ball. Figure 3 illustrates the flow of materials, energy and signals involved in carrying out this main function.

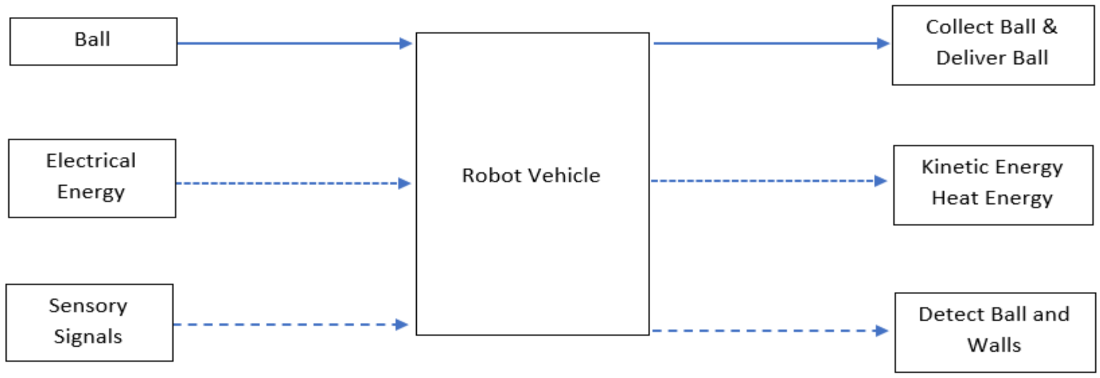


Figure 3: Main function of autonomous vehicle

Following this, the main function is broken down into smaller sub-functions to account for the intricate details in performing the primary task of ball retrieval. Supporting secondary functions are added to account for ensuring the vehicle stays within the arena boundaries and is able to avoid any obstacles (i.e. the opponent). Figure 4 shows the comprehensive functional analysis of the autonomous vehicle.

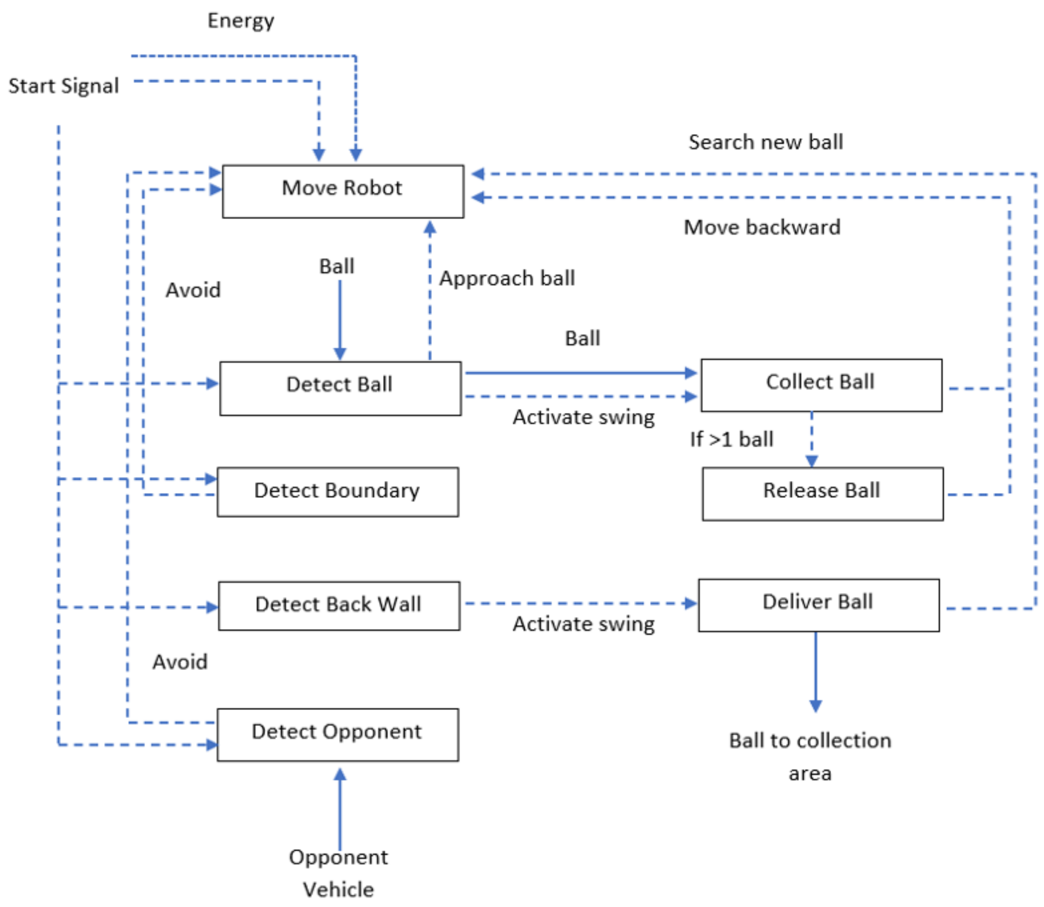


Figure 4: Detailed functional analysis of autonomous vehicle

The table below describes in detail what each sub-function entails.

Table 9: Detailed function description

Function	Description
Move Robot	✓ The robot starts by moving forward for half the arena, then scans for balls in a rotational manner ✓ When a ball is detected, the robot will approach the ball. When the ball has been collected, the robot will move backward to the collection point ✓ When the ball has been delivered, the robot will restart ball searching ✓ When the boundary is detected, the robot will retreat, and turn away from arena edge ✓ When an opponent is detected, the robot will retreat and move away from the opponent
Detect Ball	✓ Sends "approach ball" signal to Move Robot ✓ Sends "activate swing" signal to "Collect Ball"
Detect Boundary & Detect Opponent	✓ Sends "avoid" signal to Move Robot
Detect Back Wall	✓ Sends "activate swing" signal to "Deliver Ball"
Collect Ball	✓ Activate swing, rotate 90 degrees inward ✓ Sends "release" if more than one ball is detected
Release Ball	✓ Activate swing, rotate 90 degrees outward
Deliver Ball	✓ Activate swing, rotate inward all the way to kick the ball up the ramp

3.3 Morphological Chart

Following the identification of all functions, the team advanced to devise design concepts for mechanisms aimed at fulfilling each identified functional requirement. The charts below collate all the design ideas generated. The following functions will be considered: Move Robot, Collect Ball, Deliver Ball, Detect Ball, Back Wall & Opponent.

Table 10: Various configurations of wheel drive system for robot manoeuvring.

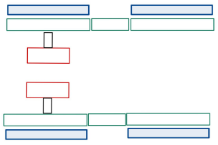
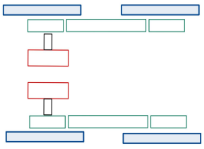
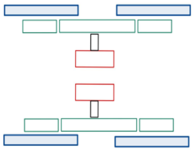
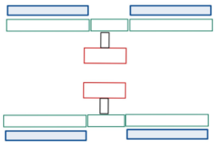
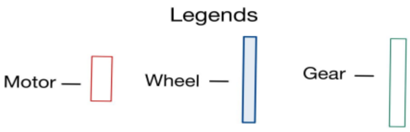
Move Robot		
1		**Rear motor drive, larger gear transmission** Pros: More weight on rear, higher traction, more room for ramp Cons: Larger turning radius
2		**Rear motor drive, smaller gear transmission** Pros: More weight on rear, higher traction, more room for ramp Cons: Larger turning radius
3		**Middle motor drive, larger gear transmission** Pros: Equal torque and speed transmission, smaller turning radius Cons: Lesser room for ramp
4		**Middle motor drive, smaller gear transmission** Pros: Equal torque and speed transmission, smaller turning radius Cons: Lesser room for ramp
		

Table 11: Various designs for ball collection mechanism.

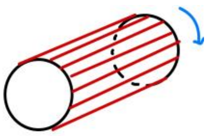
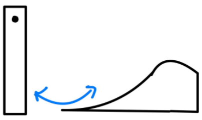
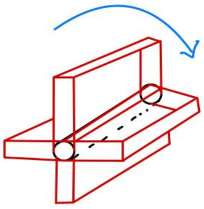
Collect Ball		
1		**Continuous roller** Pros: Smallest torque requirement, small space Cons: Possible chance of stuck ball
2		**Swinging front gate** Pros: Larger collection area, multipurpose Cons: Takes up space, possible chance of stuck ball
3		**Continuous flap** Pros: Higher chance of collection Cons: A lot of material, mass.

Table 12: Various designs for ball delivery mechanism

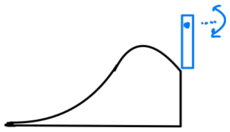
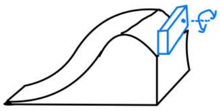
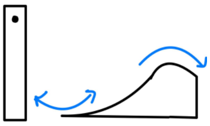
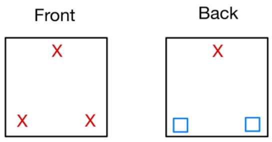
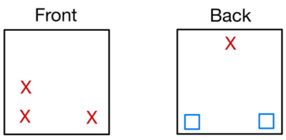
Deliver Ball		
1		**Swinging back gate** Pros: Ball is ready in place to be released Cons: Requires precise position control
2		**Rotating back gate** Pros: Ball is ready in place to be released Cons: Requires precise position control
3		**Swinging front gate** Pros: Multipurpose, one less motor. Cons: Requires precise position control, small tolerance range

Table 13: Different configuration of sensors and limit switches for object detection

Detect Ball & Opponent		
1		**2 sharp distance sensors at bottom front, 1 sharp distance sensor at top front. 1 sharp distance sensor at top back, 2 limit switches at bottom back** Pros: Cover all ranges Cons: Top sensor might not effectively scan the opponent if our vehicle is higher than the opponent.
2		**Same as 1 except the top sharp distance sensor is placed on the lower half** Pros: Can distinguish opponents more clearly.
Legends X — Sharp distance sensor □ — Limit Switch		

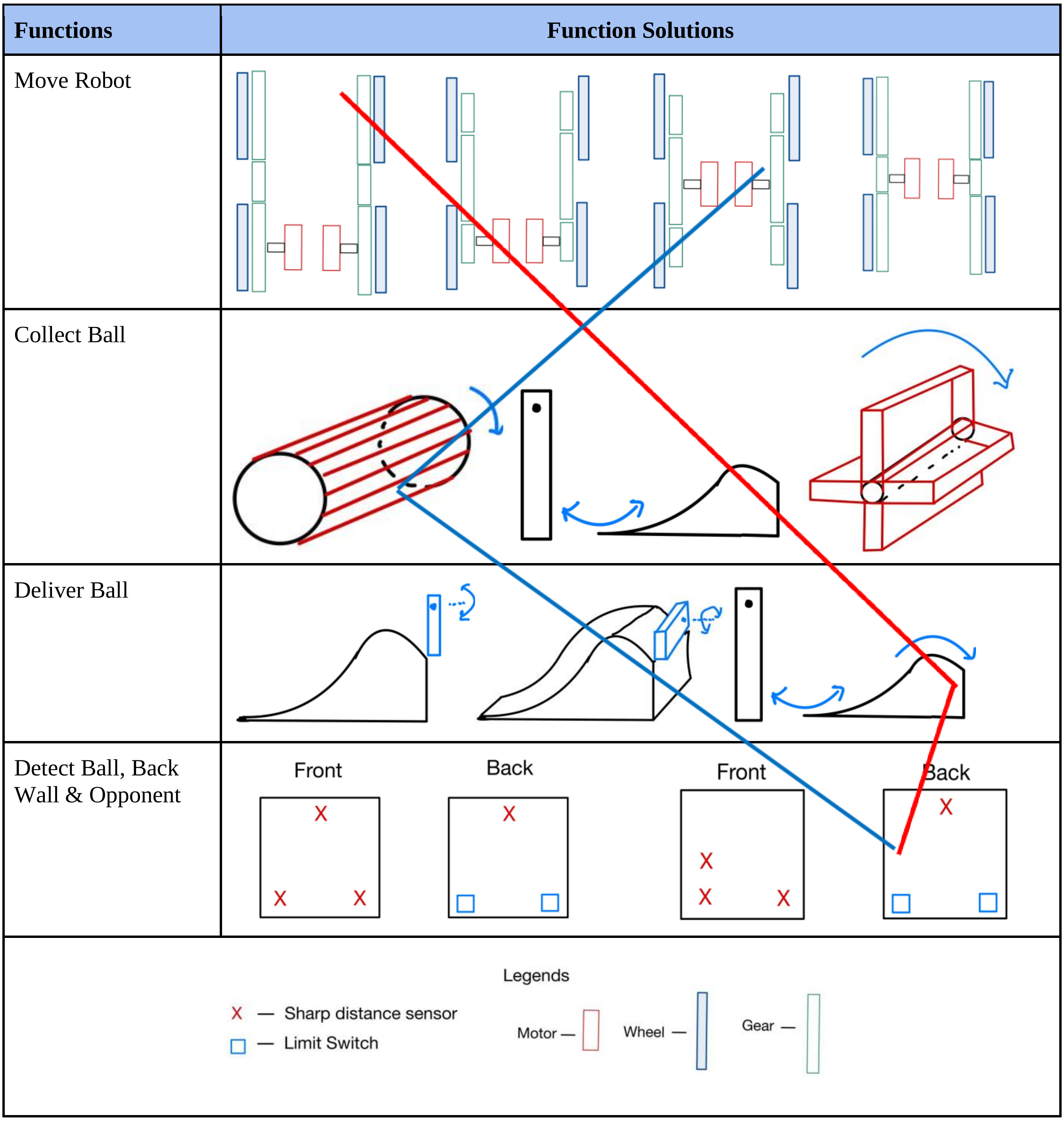


Figure 5: Morphological Chart

From the above figure, the team was able to narrow down the various ideas into two promising conceptual design ideas - namely Concept A and Concept B as indicated by the red line and blue line respectively.

Concept A

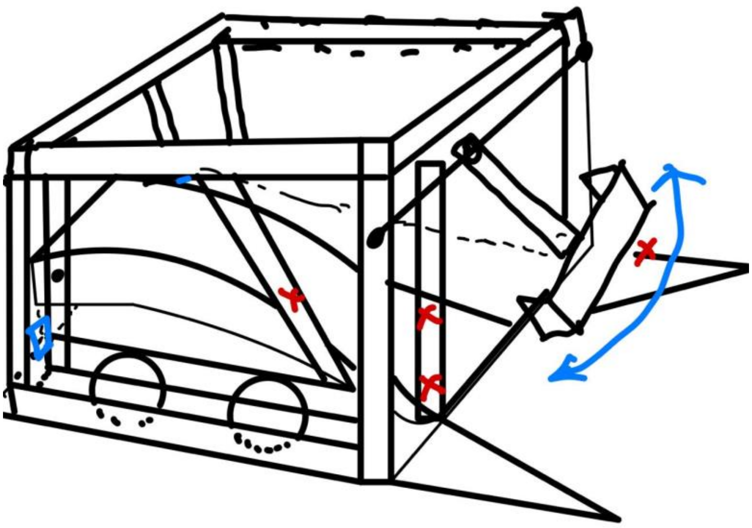


Figure 6: Concept A design.

Table 15: Description of the solutions for functions as featured in Concept A.

Sub Function	**Solution**	**Concept Description**
Move Robot	2 motors at the rear, each motor drive the big rear gear connected to a idler small gear then to a big front gear	Front & rear wheels of each side are connected via gear. Two motors each drive wheels on one side. This design synchronises the rotation of the wheel.
Detect Obstacle	Sensors	Four long distance sensors in front are employed to detect balls and opponents. Two limit switches are placed at the back to detect the ball returning area.
Collect Ball & Deliver Ball	Collector Gate	The gate will swing down to capture the ball and hold the ball on the ram. When it reaches the returning point, the gate will swing upwards and inwards the vehicle to push the ball up the ramp and over the ramp to return the ball.

Pros:

- ✓ It is stable and robust and is able to hold off collision from an opponent.
- ✓ The use of a gate collector mechanism can minimise the number of motors used for the collection mechanism.
- ✓ Simple and effective.

Cons:

- ✓ Longer turning radius results in longer turning time.
- ✓ Gate might not close if the gate force cannot overcome the friction between ball and the ramp surface.

Concept B

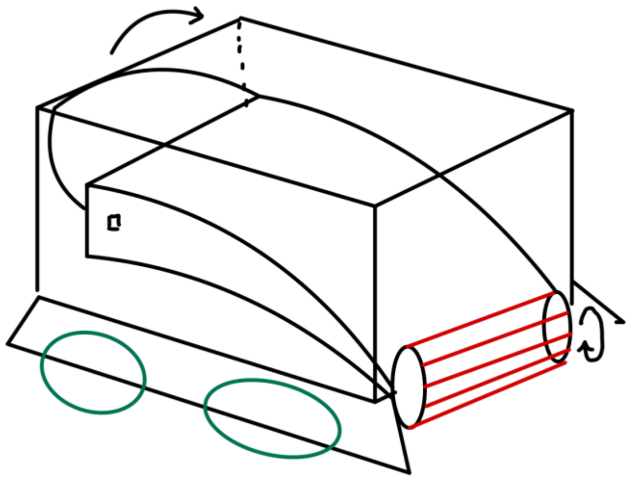


Figure 7: Concept B design.

Table 16: Description of the solutions for functions as featured in Concept B.

Sub Function	Solution	Concept Description
Move Robot	Small gear at the centre as the driving gear to drive two big gears connecting to the wheels	Using the small gear as the driving gear to drive two big gears generates high torque to the vehicle, making it more stable to handle collisions with other vehicles.
Detect Obstacle	3 sharp distance (SD) sensors are mounted vertically at the front and 2 limit switches at the back. 2 SD sensors projects to the front and another 1 project horizontally.	The vertical configuration of the SD sensors allows the robot to distinguish between the ball and the opponent, and the side SD sensor detects whether the ball is within the area to be captured. The limit switches at the back signals the robot to deliver the ball upon hitting the boundary wall.
Collect Ball	Rotating roller	Rubber wrapped around the rotating gears to provide traction to the ball. The rotation of the roller sweeps the ball up to the ramp upon contact with the ball.
Deliver Ball	Rear gate rotating sideway	The rear gate rotates sideways in a 90-degree range controlled by a motor to allow the ball to fall from the slope into the delivery box.

Pros:

- ✓ The vehicle is able to collect the ball without stopping.
- ✓ High torque generated from the wheel drive system makes it more stable and robust in collision with an opponent.
- ✓ The cylindrical-shaped roller offers a wider range of ball collection area.

Cons:

- ✓ To achieve high torque, there is a compromise in the vehicle's speed. Given that the project's goal is to retrieve balls, prioritising speed becomes crucial in the competition. Consequently, the vehicle is at a disadvantage due to its lower speed.
- ✓ The ball has a probability to be trapped at the collector.
- ✓ 4 motors add on more weight to the wheels, making the vehicle move slower.

3.4 Decision Making

After formulating conceptual ideas tailored to address specific functional requirements, the team integrated individual concepts that are compatible with one another to produce a fully functional autonomous vehicle design idea. Looking at Figure 5, two concepts were yielded. These were later assessed using a decision matrix to objectively select the most promising idea for further development.

Table 17 shows the criteria considered and weightage of each based on the design specifications described in chapter 2. Table 18 is the decision matrix the team used to choose the concept that ideally meets the desired vehicle requirements.

Table 17: Criteria Evaluation and Weightage.

Criteria	Description	Weightage
Collection & Delivery Capability	Collection chance, collection and delivery accuracy	40
Manoeuvrability	Vehicle speed, traction, path finding, efficiency, strength.	25
Collision Resistance	Ability to resist collision from another opponent, comeback from collision, and able to complete mission	20
Construction Quality	Keeping vehicle compact, neat, maintainable, sufficient	15

Table 18: Decision Matrix

Evaluation Criteria	Weighing Factor	Concept A	Concept B
Collection & Delivery Capability	40	4	4
Manoeuvrability	25	5	3
Collision Resistance	20	3	4
Construction Quality	15	5	4
Total Weight		420	375

Examining Table 18, Concept A emerged as the stronger choice over Concept B. Consequently, the team opted to advance with Concept A as the selected idea for further refinement and development.

4. Embodiment Design

After selecting the conceptual idea, the team transitioned into the Embodiment Design phase, ensuring a detailed exploration and planning of the chosen concept. This process encompasses two key facets: Embodiment Rules and Embodiment Principles. These aspects serve as guiding frameworks, allowing for a meticulous and structured approach to transform the chosen idea into a tangible and well-defined design.

4.1 Rules of Embodiment Design

The selected Concept A underwent a thorough analysis to confirm its alignment with the three foundational rules of embodiment design: Clarity, Simplicity, and Safety. This scrutiny ensured that the chosen concept adhered to these essential principles, improving the design so that it is clear, and straightforward, and prioritises safety considerations.

4.1.1 Clarity

To fulfil the rule of clarity, the details and functionality of the vehicle design should be clear and straightforward, ensuring they are easily understandable to facilitate ease in construction and repair. The chosen vehicle design includes several explicit functions as follows to ensure this clear understanding of what each component of the vehicle is for:

1. Wheels + Gears: Vehicle manoeuvring and power transmission
2. External Frames: Provide mounting points, platform and safeguard the components onboard
3. Collector Mechanism: Collect the ball and secure it on the ramp
4. Ramp: Path for the ball into collection area during delivery process
5. Battery: Provide power supply to the robot
6. Line tracking module: Edge detection
7. Limit switches: Determine if the ball is delivered
8. Distance sensors: Detecting ball and opponents, confirms the number of balls
9. Digital compass: Provide orientation information to the robot

4.1.2 Simplicity

Simplicity in design should be considered in terms of both hardware and software. In hardware design, simplicity refers to the principle of creating an uncomplicated hardware system with easy installation of parts and sensors, without compromising the robot's functionality or performance. Likewise, simplicity in software design involves the decomposition of a complicated system into multiple simple tasks, and linking them up to form a complete system.

In terms of hardware, the chosen vehicle design displays a modular architecture. Critical components such as the gate and ramp can easily be assembled or disassembled swiftly. Additionally, each component in the vehicle design has an integral role for the vehicle’s overall functionality, thus reducing the use of redundant parts.

The team classified the vehicle’s operation into two main tasks: robot navigation and ball retrieval. These two main tasks can be further broken down into many simple functions. Robot navigation comprises robot manoeuvring and edge detection while the ball retrieval system involves ball searching, ball capturing, and ball delivery. A comprehensive program flowchart was created to provide an overview of the system to improve the programming efficiency. By doing so, it is easy to understand the flow of logic in the vehicle operation.

4.1.3 Safety

Safety is of utmost importance in embodiment design. The robot should not pose any threat to the components interacting with it, nor to other vehicles and people in the surrounding area. Safety must always be the primary consideration in mechatronics design to minimise risks and ensure safe operation. The following measures have been implemented to ensure safety of the vehicle:

1. Spacers and collars are used between the wheels and gears to secure them in place.
2. Insulation of the IR tracking module pins using tape to prevent direct contact with the metal bar, which could lead to a short circuit of the device.
3. Sensors are protected by metal plates and 3D-printed cases to prevent any damage from collisions during the competition.
4. The drive system is enclosed within a metal frame to prevent direct collision with an opponent's vehicle onto the gears.
5. Increased infill intensity for the 3D printing of the collector gate to prevent it from being damaged easily.

4.2 Principles of Embodiment Design

Examining ConceptA, the team verified its adherence to the core principles of embodiment design: Force Transmission, Division of Tasks, Self-Help, Stability, and Instability. This analysis guarantees that the chosen design idea will ultimately yield an effective vehicle, aligned with good engineering practices. This approach ensures that the vehicle is not only well-designed but also engineered to meet high standards of efficiency, functionality, and reliability.

4.2.1 Force Transmission

Emphasising the Principle of Direct and Shortest Transmission Path, the motor's torque is directly transmitted to both the gear and the rear driving wheel. This strategic transmission ensures an efficient and direct power transfer, contributing to the overall effectiveness and performance of the vehicle.

4.2.2 Division of Task

Every component of the vehicle has a specific function, and works together seamlessly to create a fully integrated autonomous system. The drive system is responsible for the robot's manoeuvrability, allowing it to move and position itself as needed. The sharp distance sensors play a crucial role in detecting tennis balls, ensuring the robot can locate and interact with the tennis ball effectively. The collector gate and ramp are key to the ball collection and delivery mechanisms, handling the capture and transport of the balls to the delivery box.

4.2.3 Self-Help

The vehicle's ramp is positioned to make contact with the ground through the influence of gravitational force. This design ensures a self-sufficient and gravity-driven mechanism, contributing to the vehicle's operational simplicity and efficiency.

4.2.4 Stability

The wheels are placed symmetrically to provide stability to the robot motion. The VEX microcontroller and the battery pack are spread out on the roof in such a way to evenly distribute the weight onto the wheels.

5. Detailed Design

This chapter provides an in-depth exploration of the selected concept, Concept A. It begins by offering a comprehensive justification of the selected design, detailing the implementation, and enhancements for each hardware element, including the overall Structure, Drive system, and the Ball Collection Mechanism. Subsequently, it delves into the calculations associated with the drive system and retrieval mechanism. The focus then shifts to the overall assembly, incorporating detailed engineering drawings of individual parts. The assembly process of each electrical component such as Motors, Sensors, and Limit Switch is also outlined in detail. To conclude, a comprehensive wiring diagram encapsulating the entirety of the electrical system is shown.

5.1 Hardware Design

5.1.1 Chassis

The main priority is to limit the chassis size to 30 cm in length and 30 cm in width. The vehicle is constructed with a square base of () characterised by a hollow structure crafted from metal plates and bars to accommodate a 3D-printed ramp. To ensure collision prevention, wheels, motors, and gears were placed strategically within the confines of the base frame. The ball collector gate is placed at a height of ___. When the gate swings to collect the ball, its collection seamlessly adheres to the curvature of the ramp. The placement of the battery and VEX controller were initially meant to be under the ramp. However, due to spatial constraints, the mentioned component was relocated to the vehicle's top plate.

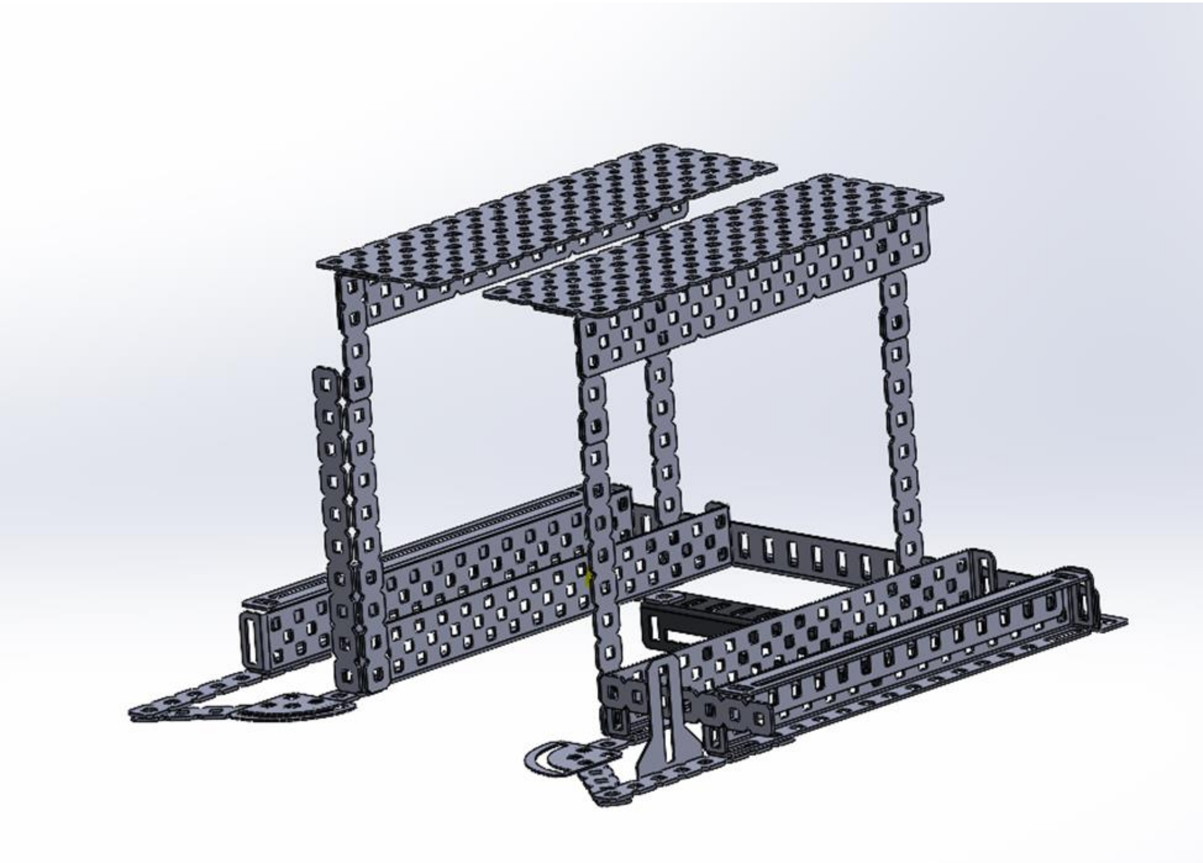


Figure 8: Final chassis design of the vehicle.

5.1.2 Drive System

The primary concerns were centred around achieving optimal manoeuvrability and collision resistance, both of which align with our objective of effectively collecting and delivering the tennis ball.

Manoeuvrability, in this context, involves precise control over speed and turning speed. To achieve this, the implementation of a synchronised rotation mechanism for the two wheels on the same sides by employing gears which also ensures they rotate at the correct speed. By controlling the manoeuvrability with just two motors, the vehicle turning speed is effectively managed which is akin to a differential drive system.

Collision resistance, on the other hand, refers to the vehicle's capability to maintain its intended path even in the event of a collision with the opponent's vehicle. This is facilitated by the complete enclosure of the wheel and gear system within the base frame, ensuring robust protection and resilience during collisions.

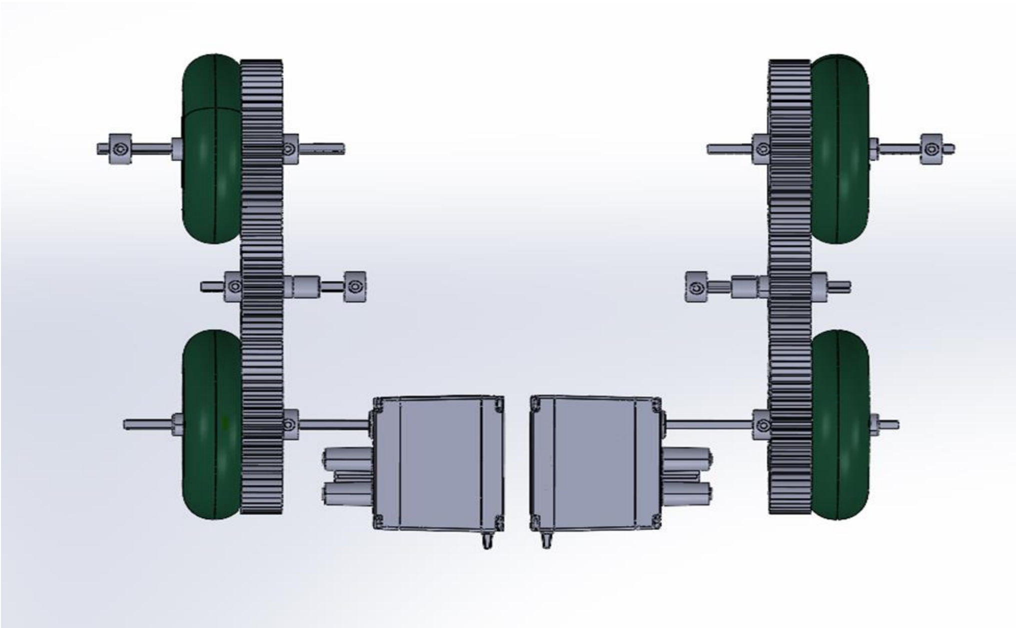


Figure 9: Drive system of the vehicle (top view).

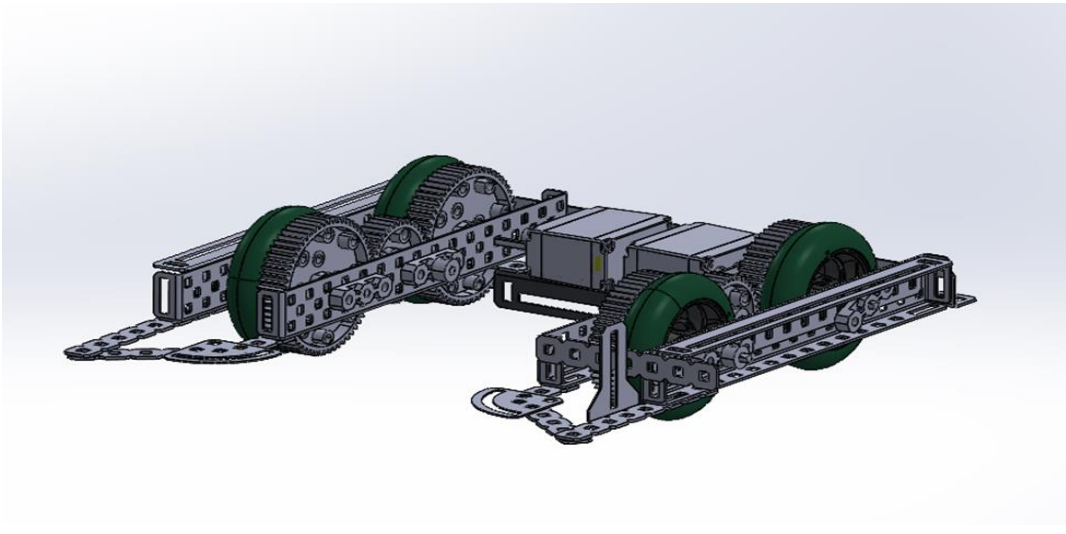


Figure 10: Drive system of the vehicle (isometric view).

5.1.3 Ball Collection Mechanism

This section elaborates on the strategies and executions employed by the vehicle for the collection of tennis balls which are placed randomly in the arena. It also outlines the process of depositing the collected tennis ball into the designated collection area within the arena. The collection mechanism is intricately divided into two distinct parts: The Ball Collector Gate and the Ramp which are designed to guide the ball into the vehicle and the swinging of the gate upward causes the ball to go up the ramp and deposit into the designated collection area.

Ball Collector Design

The ball collector resembles a rotatable swing gate with an overall dimension of 9.8 cm by 2cm by 10.8cm (L x W x H) installed at the front of the vehicle. It consists of 3 modular parts: the swings, the base, and the lock, which are 3D printed in Acrylonitrile butadiene styrene (ABS) polymer material. The modular design aims to save time in 3D printing and also offers ease of repair and replacement in case of any damaged parts during testing and competition.

The first design, also the final design (see Figure xx) of the swing is designed to be 8.5 cm long and has a 3.2 mm long square hole to allow a 17 cm long metal shaft to pass through. The shaft is connected at both ends to a continuous motor and a shaft encoder to rotate the collector gate. The bottom of the swing features an extruded part designed to fit into the hole on the modular base. This design generally performed well during vehicle performance testing. However, there were instances where the ball was trapped under the gate, leading the motor to stall and a halt in vehicle operation.

Troubleshooting revealed that the motor lacked sufficient torque to push the ball into the vehicle without the addition of counterweights. Thus, the second design incorporates an extrusion from the rotation axis to accommodate metal bars as counterweights, aiming to minimise the torque needed to rotate the gate. However, this design underperformed compared to the first because the moment of inertia of the second design is smaller, given $\tau = I\alpha$, torque is smaller, causing it to not be able to provide enough force and acceleration to the ball up the ramp. The solution to the problem that surfaced in the first design involves moving the vehicle forward during the ball-capturing process and adding metal gussets on both sides of the front chassis. The gusset with its curvy shape, allows the ball to slide smoothly to the middle of the gate. Therefore, the first design is used as the final design.

The modular base is 10 cm long and 1.9 cm tall with 1.5 cm extruded walls on both sides (see Figure 13) to prevent the ball from sliding out of the gate during the ball-capturing process. It is designed to have two 10.8 mm by 6.8 mm square holes to accommodate two swing parts. These modular parts are secured with two 6 mm by 6 mm locks to complete the assembly.

Table 19: Illustration of the modular parts and the problems associated with various swing designs.

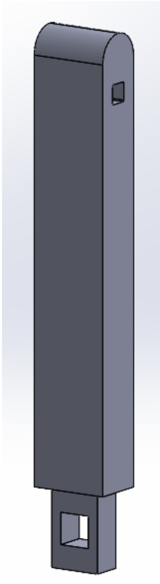
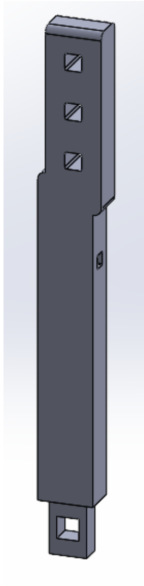
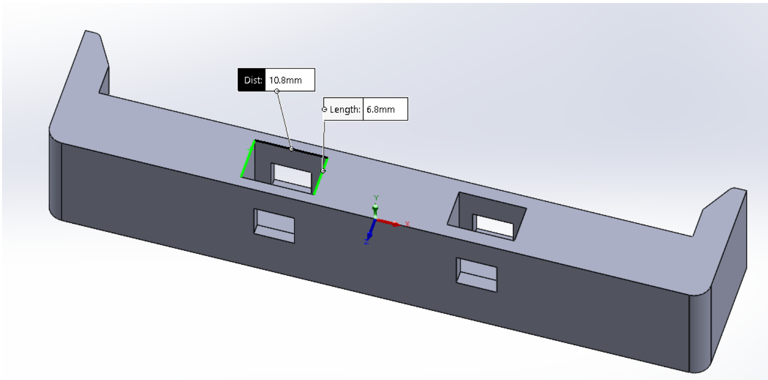
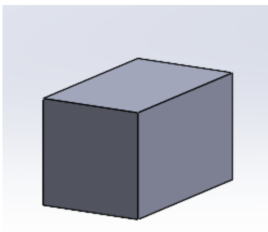
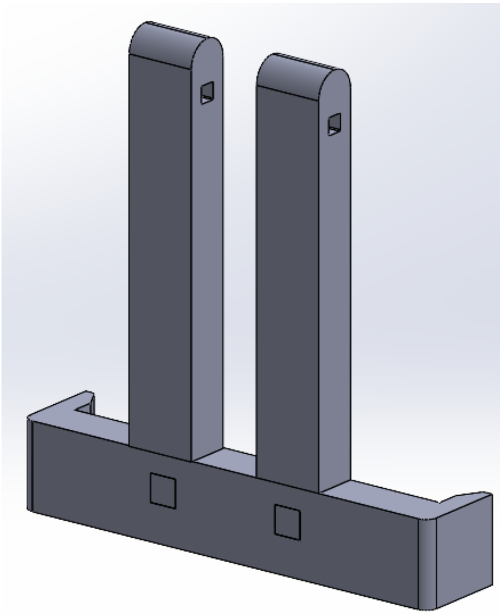
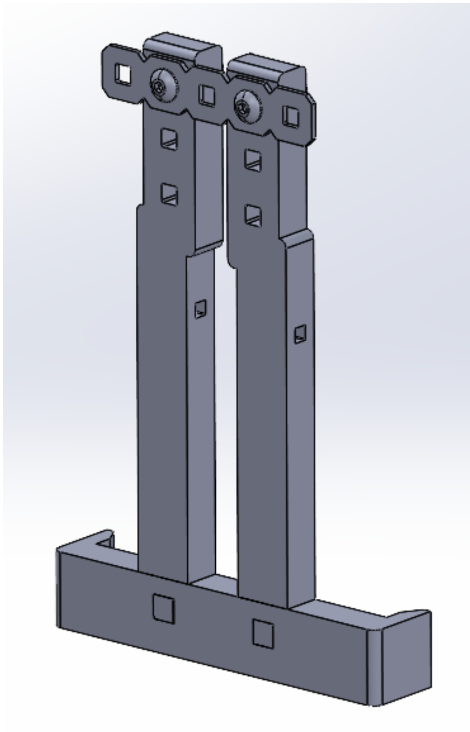
<table>
<tr><th>Part</th><th>First (Final) Design</th><th>Second Design</th></tr>
<tr><td>Swing</td><td>
Figure 11: Modular swing first design.</td><td>
Figure 12: Modular swing second design.</td></tr>
<tr><td>Problem</td><td>Ball sometimes trapped under the gate</td><td>Underperformed compared to first design</td></tr>
<tr><td>Base</td><td colspan="2">

Figure 13: Modular base design</td></tr>
<tr><td>Lock</td><td colspan="2">
Figure 14: Modular lock design</td></tr>
</table>

<table>
<tr><td>Assembly</td><td>
Figure 15: Assembly of the gate without counterweight.

Final Design ✓</td><td>
Figure 16: Assembly of the gate with counterweight.</td></tr>
<tr><td>Solution to problem</td><td colspan="2">Accelerate the vehicle forward during the ball-capturing process and add metal gussets on both sides of the front chassis to slide the ball to the middle of the gate, using its curvy shape.</td></tr>
</table>

Ramp Design

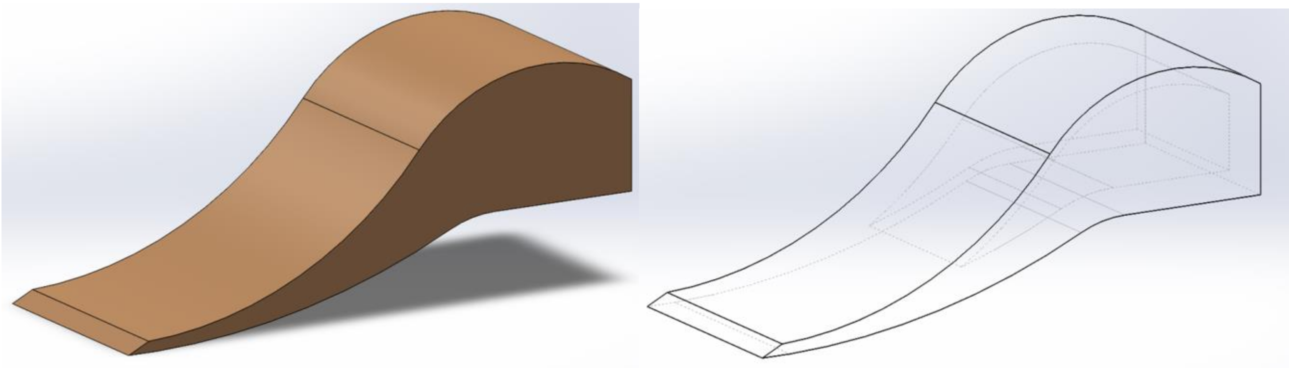


Figure 17: Solid (left) and hollow (right) view of the ramp design.

Table 20: Advantages of the features in ramp design.

Features	Advantages
Hollow Shell	Reduce mass, reduce 3D printing time
Small base surface are	Reduce friction
Slanted ramp at start of ramp	Increase ramp rigidity, reduce chance of ramp wreckage or buckling
Curved ramp	Allow circular gate motion
Smooth curvature	Smooth ball path
Increasing sectional area	In case of horizontal stress on the ramp, reduce stress concentration
Empty bottom space at ramp end	Allows placement of motors

5.1.4 Ball Delivery Mechanism

The ball delivery mechanism refers to the process of unloading the ball from the vehicle into the delivery box when the vehicle is within the delivery area. Our ball delivery mechanism is designed to use the collector gate, by rotating it backward to push the ball up the ramp and allow it to fall from the back. This action is triggered by limit switches installed at the back, which activate when pressed against the boundary wall. A sharp distance sensor is mounted inside the vehicle to detect the presence of a ball captured by the gate. Once the ball is collected, the vehicle adjusts its position so that its back faces the delivery area, and then it reverses into the delivery area to deliver the ball.

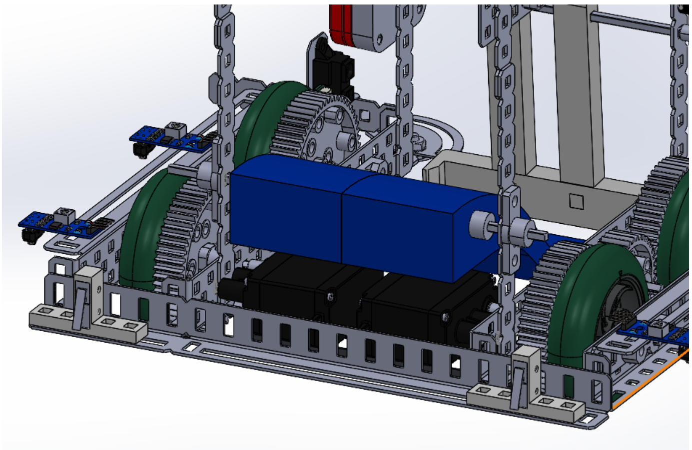


Figure 18: CAD modelling of the ball delivery system.

5.2 Design Calculation

5.2.1 Power Consumption

Motor Specification

Table 21 summarises the specifications of the two motors used for the vehicle. The information is obtained from the datasheet shown in Appendix X. The 2-wire Motor has two configurations “High Speed” or “Turbo” modes. For this project, the “High Speed” configuration was used.

Table 21: Motor Specifications.

Motors Type	Free Speed(rpm)	Stall Torque (Nm)	Free Current (A)	Stall Current (A)
3 Wire Motor (276-2163)	100 at 7.5V	0.7344 (6.5 in-lbs)	0.14	1.6
2 Wire Motor 393 (High Speed Option)	160 at 7.2V	1.05 (8.4 in-lbs)	0.37	4.8

Both motors can only operate in continuous mode. The project uses the 3-wire motor for the ball collector gate and the two 2-wire motors were used to drive the vehicle. Two motor controllers (Motor Controller 29) were used, one each for each 2-wire motor.

Power System

With three rounds of the competition, with each round lasting 3 mins, the following calculations are performed to ensure that there is sufficient battery power to operate the robot throughout the competition.

$$Total\ duration\ of\ competition\ =\ 3\ rounds\ \times\ 3\ mins\ =\ 9\ mins$$

Taking into consideration of trial runs, debugging runs and additional rounds, it is assumed that the robot will operate for 15 mins with a fully charged battery (3000mAh, 7.2V):

$$Battery\ power\ =\ \frac{3000mAh\ \times\ 10^{-3} \times 7.2V}{\frac{15\ mins}{60}} = 86.4W$$

The following table tabulates the power used by each electrical component

Table 22: Power consumption of different components.

	No	Current Consumption (A)	Voltage (V)	Power Consumed per device (W)	Total Power Consumed (W)
2-Wire Motor 393	2	0.37	7.2	2.66	5.32
3-Wire Motor (276-2163)	2	0.14	7.5	1.05	2.10
Motor Controller	2	0.012	8.5	0.10	0.20
Sharp Distance Sensor (Long)	3	0.030	5.0	0.15	0.45
Sharp Distance Sensor (short)	1	0.012	5.0	0.06	0.06
IR Line Tracking Module	4	0.020	5.0	0.10	0.40
Digital Compass	1	0.020	5.0	0.10	0.10

$$Total\ Power\ Consumption\ \ =\ 5.32\ +\ 2.10\ +\ 0.20\ +\ 0.45\ +\ 0.06\ +\ 0.10\ =\ 8.63W$$

Since total power consumption is less than the battery power provided, the vehicle is operating well within the scope of the battery life.

5.2.2 Drive System Force Analysis

Table xx summarises the mass of the components of the robot, to facilitate the calculation of force and torque needed to drive the robot.

Table 23: Mass of individual components of the robot.

Component	No.	Mass (kg)	Total Mass (kg)
Vex Microcontroller	1	0.137	0.137
Motor Controller 29	2	0.009071847	0.00907
Battery (276-1491)	1	0.312	0.312
2-Wire Motor 393	2	0.087089735	0.174
3-Wire Motor (276-2163)	1	0.0952544	0.0953
Gears(60T and 36T)	6	0.04088	0.0409
Wheels	4	0.05	0.2
Chassis*	N/A	2	2
3D Printed Collector (Gate)	1	0.04429	0.0443
3D Printed Ramp	1	0.19587	0.196
3D Printed Limit Switch	2	0.00251	0.00502
Sharp Distance Sensor	4	0.015	0.06
Sensors (IR Line Tracking Module, Limit Switches, Compass)*	N/A	0.090	0.09
Others (Screws, Nuts, Cables etc.)*	N/A	0.050	0.050
Tennis Ball	1	0.058	0.058

**The mass of the components is a rough estimate for calculations purposes*

$$Robot\ mass\ =\ sum\ of\ the\ mass\ of\ the\ components\ =\ 3.47159\ kg$$

Force Analysis

$$mass\ of\ vehicle, m\ =\ 3.73kg\ , radius\ of\ the\ wheel, r\ =\ 3.5\times\ 10^{-2}\ m$$
$$radius\ of\ the\ wheel, r\ =\ 3.5\times\ 10^{-2}\ m$$

To measure coefficient of friction between the wheel and the arena surface, we could make use of the equation:

$$\mu = tan\ \theta$$

where θ is the angle of the inclined surface until the wheel starts sliding down the surface.

Since it was not possible to experiment with the arena surface, an approximation of 0.5 is taken for the coefficient of friction.

$$coefficient\ of\ friction\ between\ the\ rubber\ wheel\ and\ arena\ surface, \mu\ = 0.5$$

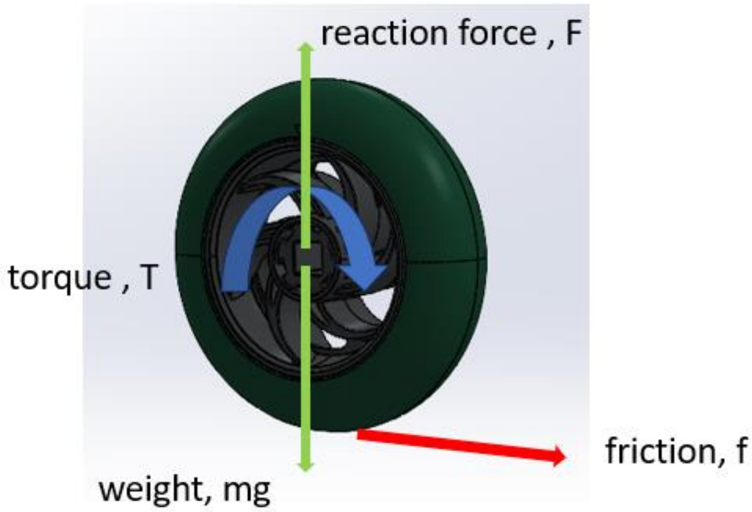


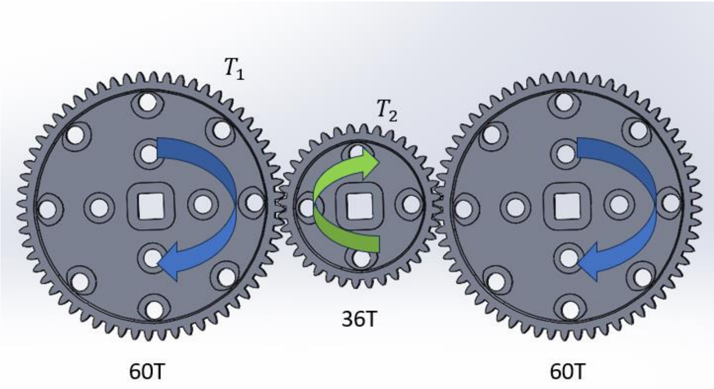
Figure 19: Force analysis of wheel

Being a 4-wheel drive system, the weight of the robot is assumed to be evenly distributed among 4 wheels:

$$Reaction\ force, F\ =\ mg\ =\ \frac{3.47}{4}\ kg\ \times\ 9.81\ m\, s^{-2} =\ 8.51\ N$$
$$Friction\ force, f\ =\ \mu F\ =\ 0.5\ \times\ 8.51\ N\ =\ 4.255\ N$$
$$Torque\ required\ to\ drive\ the\ wheel\ , T\ =\ f\ \times\ r\ =\ 4.255\ N\ \times\ 3.5\times\ 10^{-2}\ m\ =\ 0.149\ Nm$$

Torque to Drive the Wheel

The following diagram shows one side of the gear system of the drive system.



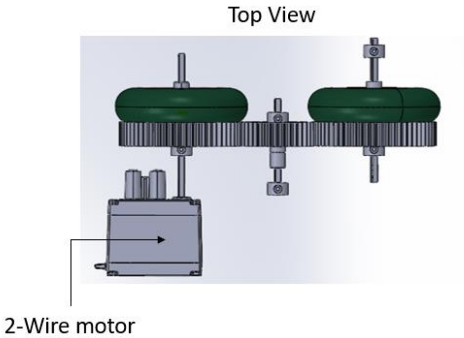


Figure 20: Gear train of the wheel drive system.

Observing from the top view of the gear train, the 60T gear on the left is connected to the 2-wire motor. It is assumed that the gear train has an efficiency of 90%.

$$Gear\ Ratio\ = \frac{n_3}{n_2}\frac{n_2}{n_1}$$
$$= \frac{n_3}{n_1} = \frac{60}{60} = 1$$

Under constant velocity operation, the torque needed for the motor is given below.

$$T_1 = 0.149\ Nm$$

With 90% efficiency, the torque needed for the motor, $T_{motor} = \frac{0.149}{0.9} = 0.166\ Nm$

<u>Under acceleration operation:</u>

For the competition, we operate the robot to drive at the speed of $xx\ m\ s^{-1}$
Hence, $\omega_{motor} = 84.19\ rpm.$ **(refer to Robot Operating Speed for detailed calculation)**
Assuming the motor accelerate to the angular velocity of $84.19\ rpm$ **within 0.5s**

$$angular\ acceleration, \alpha = \frac{84.19 \times \frac{2\pi}{60}}{0.5} = 17.63\ rad\ s^{-1}$$
$$T_{acceleration} = J\alpha$$

To calculate the moment of inertia , J:

The screws, lock nuts and shafts are assumed to have negligible mass. The following table summarises the moment of inertia, J of each component. The values for J were obtained from SolidWorks which were imported from the Vex Website (refer to appendix A):

Table 24: Moment of inertia of each component of the drive system.

Components	Moment of Inertia, $J\ g\ mm^2$	Moment of Inertia, $J\ kg\ m^2$
36T Gear	939.30	9.398×10^{-7}
60T Gear	5263.13	5.261×10^{-6}
Wheel	28972.69	2.897×10^{-5}
2-Wire Motor	39164.57	3.916×10^{-5}

The moment of inertia on driven gear mechanism is in the ratio of the inverse square of the gear ration, r when seen from the driving gear mechanism, and the efficiency η of the gear mechanism must also be considered.

$$J_{reflected} = \frac{J}{r^2 \eta}$$

To calculate the moment of inertia , J :

$$J = J_{motor} + J_{60T\ Gear} + J_{wheel} + J_{36T\ Gear\ ,reflected} + J_{60T\ Gear,reflected} + J_{wheel;,reflected}$$

$$J = 3.916 \times 10^{-5} + 5.261 \times 10^{-6} + 2.897 \times 10^{-5} + \frac{9.398 \times 10^{-7}}{0.9(\frac{36}{60})^2} + \frac{5.261 \times 10^{-6}}{0.9^2(1)^2} + \frac{2.897 \times 10^{-6}}{0.9^2(1)^2}$$

$$J = 8.636 \times 10^{-5}\ kg\ m^2$$

Hence, to calculate the required motor torque to accelerate:

$$T_{acceleration} = J\alpha = 8.636 \times 10^{-5} \times 17.63 = 0.00152\ Nm$$

Total torque required by the motor:

$$T_{total} = T_{acceleration} + T_{steady\ state} = 0.00152 + 0.166 = 0.16752\ Nm$$

Under breakaway operation:

The motor shaft is connected to the load with sleeve bearings and the motor has been used regularly, the breakaway torque is assumed to be 140% of the running torque:

$$Breakaway\ Torque = 1.4 \times 0.166 = 0.232\ Nm$$

Conclusion:

Breakaway	Acceleration	Running(Steady State)
$0.232Nm$	$0.167Nm$	$0.166\ Nm$

Since $0.232Nm \ll 1.05Nm\ (Stall\ torque\ of\ 2-wire\ motor)$, the 2-wire Motor is a good fit for the drive system.

<u>Robot Operating Speed</u>

With known stall torque ($1.05\ Nm$) and free angular speed $160\ rpm$ of the 2-wire motor. A linear graph of torque against angular velocity is plotted as shown in Figure 21.

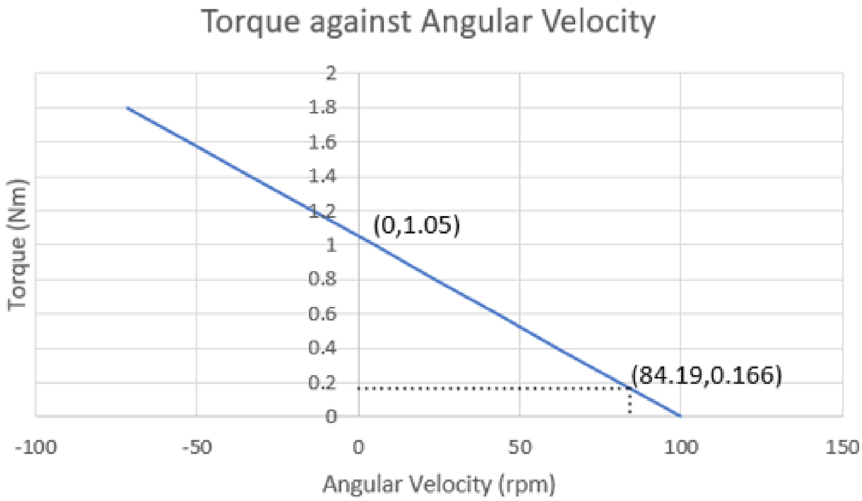


Figure 21: Graph of torque against angular velocity for 2-wire motor

$$Linear\ equation: T = -0.0105\omega + 1.05$$
$$At\ T_{motor} = 0.166\ Nm, \omega_{motor} = \omega_{60T} = 84.19\ rpm$$
$$Gear\ Ratio = \frac{N_{60T}}{N_{36T}} = \frac{\omega_{60T}}{\omega_{36T}} = \frac{60}{36} = \frac{5}{3}$$
$$\frac{5}{3} = \frac{\omega_{60T}}{\omega_{36T}} = \frac{84.19}{\omega_{36T}}$$
$$\omega_{36T} = 50.51\ rpm$$
$$\omega_{wheel} = \omega_{60T} = 84.19\ rpm\ , r_{wheel} = 3.5 \times 10^{-2}m$$
$$velocity, v = r_{wheel} \times \omega_{wheel} = \frac{84.19\ rpm \times 2\pi}{60} \times 3.5 \times 10^{-2}m = 0.309\ m\ s^{-1}$$

Since the competition area is $2.4m \times 1.2m$ **,**

$$v = \frac{distance, d}{time, t}$$
$$0.309 = \frac{2(2.4)}{t}$$
$$t = 15.53\ s$$

The vehicle is capable of making a round trip from the start point of the arena to the end of the arena and back to the start point again within 20s. This is deemed sufficient for the successful retrieval of the tennis ball under the ideal condition of required retrieval of at least 2 times for the collection of 2 balls in total, estimated to take only 31.06s which is under 3 minutes.

5.2.3 Ball Collection Force Analysis

Force Analysis

The following assumptions are made for the force analysis of the ball up the ramp:

1. The ball and the gate does not deform when compressed
2. There is always only one point of contact between the ball and the ramp
3. The ball does not slips as it rolls
4. During the first contact between the ball and the ramp, the centre of the roller, the bottom tip of the ramp and the centre of the ball forms a straight line (refer to figure xx below)
5. Given that the curvature of the ramp has a very big radius, the ramp is approximated as a slope with constant thete, θ
6. The coefficient of friction between the roller and the 3D-printed ramp is $\mu = 0.60$
7. Radius of the ball is 33mm, the centre of the ramp curve is 150mm away.

The figure below shows the force analysis of the ball rolling up the ramp

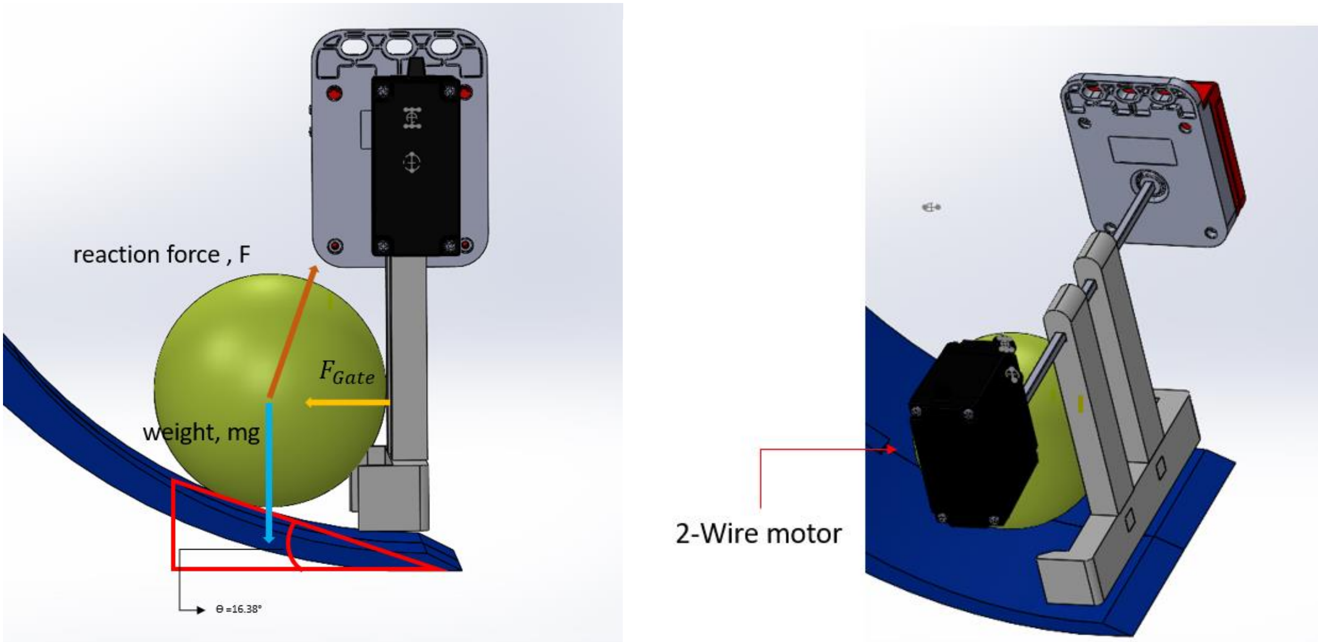


Figure 22: Force analysis of tennis ball up the ramp

$$friction, f = \mu R = 0.6R$$
$$mass\ of\ tennis\ ball, m_{ball} = 58 \times 10^{-3} kg$$
$$weight\ of\ tennis\ ball, W_{ball} = 58 \times 10^{-3} \times 9.81 = 0.569N$$

Consider Horizontal Component of the forces:

$$\sin\theta = \frac{33}{150 - 33}$$
$$\theta = 16.38°$$
$$\mu = 0.6$$
$$F_{gate} = R\sin\theta + f\cos\theta = R\sin\theta + \mu R\cos\theta = R(\sin\theta + \mu\cos\theta)$$
$$F_{gate} = R(\sin 16.38 + (0.6)\cos 16.38) \quad ----(1)$$

Consider Vertical Component of the forces:

$$Rcos\theta \ = \ fsin\theta \ + \ W_{ball}$$
$$Rcos\ 16.38 \ = \ (0.6)(R)sin\ 16.38 \ + 0.569 ----(2)$$

By solving (1) & (2):

$$R \ = \frac{0.569}{cos\ 16.38 \ - \ (0.6)sin\ 16.38} = 0.720N$$
$$F_{gate} \ = \ 0.720(sin\ 16.38 \ + \ (0.6)\ cos\ 16.38) \ = \ 0.894N$$
$$T_{gate} \ = \ F_{gate} \times r \ = \ 0.894 \ \times \ 112.25 \ \times \ 10^{-3} \ = \ 0.100\ Nm$$

Considering 90% efficiency,

$$T_{motor} \ = \ \frac{0.100}{0.9} \ = \ 0.09\ Nm$$

<u>Torque of drive the collector gate</u>

Under constant velocity operation:

$$T_{motor} \ = \ 0.09\ Nm$$

Under Acceleration operation:

During the competition, the 3-wire is driven at an angular velocity,

$\omega_{motor} \ = 87.745\ rpm$ (refer to gate collect motor speed)

Assuming the motor accelerate to the angular velocity of $84.19\ rpm$ **within 0.5s**

$$angular\ acceleration, \alpha \ = \ \frac{87.745 \ \times \ \frac{2\pi}{60}}{0.5} = \ 18.38\ rad\ s^{-1}$$
$$T_{acceleration} = J\alpha$$

To calculate the moment of inertia, J**:**

The screws, lock nuts and shafts are assumed to have negligible mass. The following table summarises the moment of inertia, J of each component. The values for J were obtained from SolidWorks which were imported from the Vex Website (refer to appendix A):

Table 25: Moment of inertia of the collector gate and motor.

Components	Moment of Inertia, $J\ g\ mm^2$	Moment of Inertia, $J\ kg\ m^2$
Collector Gate	24546.48	$2.455 \ \times \ 10^{-5}$
3-Wire Motor	31842	$3.1842 \ \times \ 10^{-5}$

$$J = J_{motor} + J_{Collector\ Gate}$$

$$J = 3.1842 \times 10^{-5} + 2.455 \times 10^{-5} = 5.6392 \times 10^{-5}\ kg\ m^2$$

$$T_{acceleration} = J\alpha = 5.6392 \times 10^{-5} \times 18.38 = 0.00104\ Nm$$

Total torque required by the motor:

$$T_{total} = T_{acceleration} + T_{steady\ state} = 0.00104 + 0.09 = 0.09104\ Nm$$

Under breakaway operation:

The motor shaft is connected to the load with sleeve bearings and the motor has been used regularly, the breakaway torque is assumed to be 140% of the running torque:

$$Breakaway\ Torque = 1.4 \times 0.09 = 0.126Nm$$

Conclusion:

Breakaway	Acceleration	Running (Steady State)
$0.126Nm$	$0.09104Nm$	$0.09\ Nm$

Since $0.126Nm \ll 0.7344\ Nm\ (Stall\ torque\ of\ 3 - wire\ motor)$, the 3-wire Motor is a good fit for the collector system.

<u>Gate Collector Motor Speed</u>

With known stall torque ($0.7344\ Nm$) and free angular speed ($100\ rpm$) of the 3-wire motor. A linear graph of torque against angular velocity is plotted as shown in Figure 24.

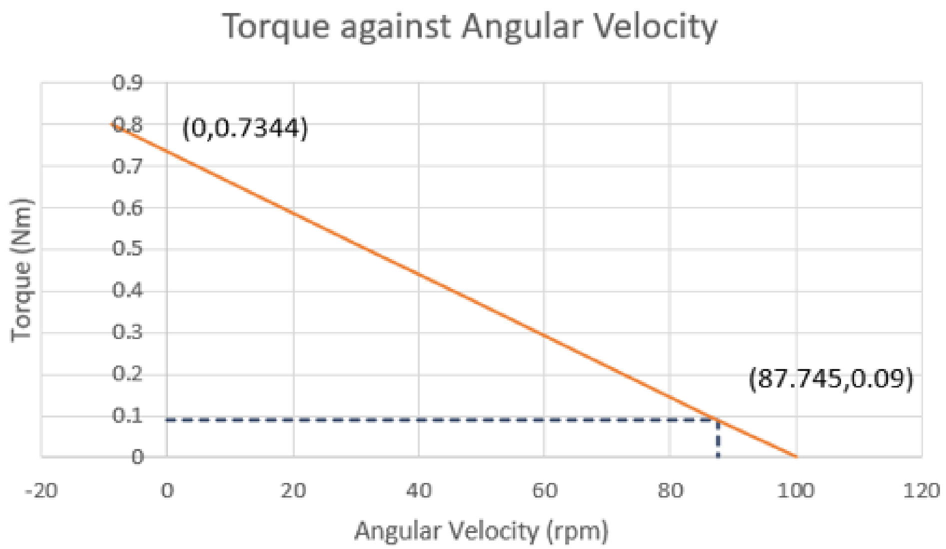


Figure 23: Graph of torque against angular velocity.

$$Linear\ equation:\ T = -0.0105\omega + 1.05$$

$$At\ T_{motor} = 0.09Nm, \omega_{motor} = 87.745\ rpm$$

5.3 Engineering Drawings

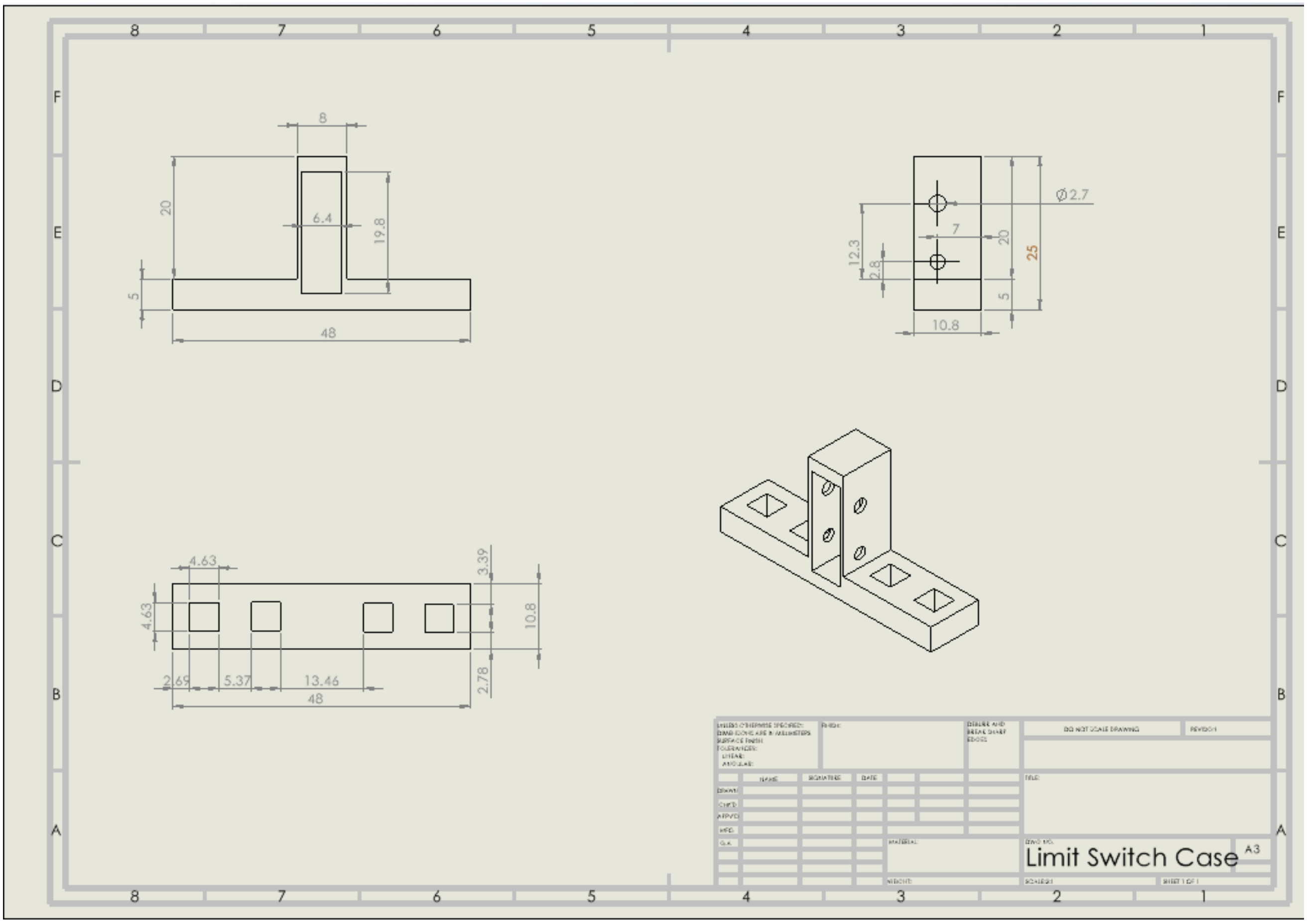


Figure 24: Limit switch case drawing

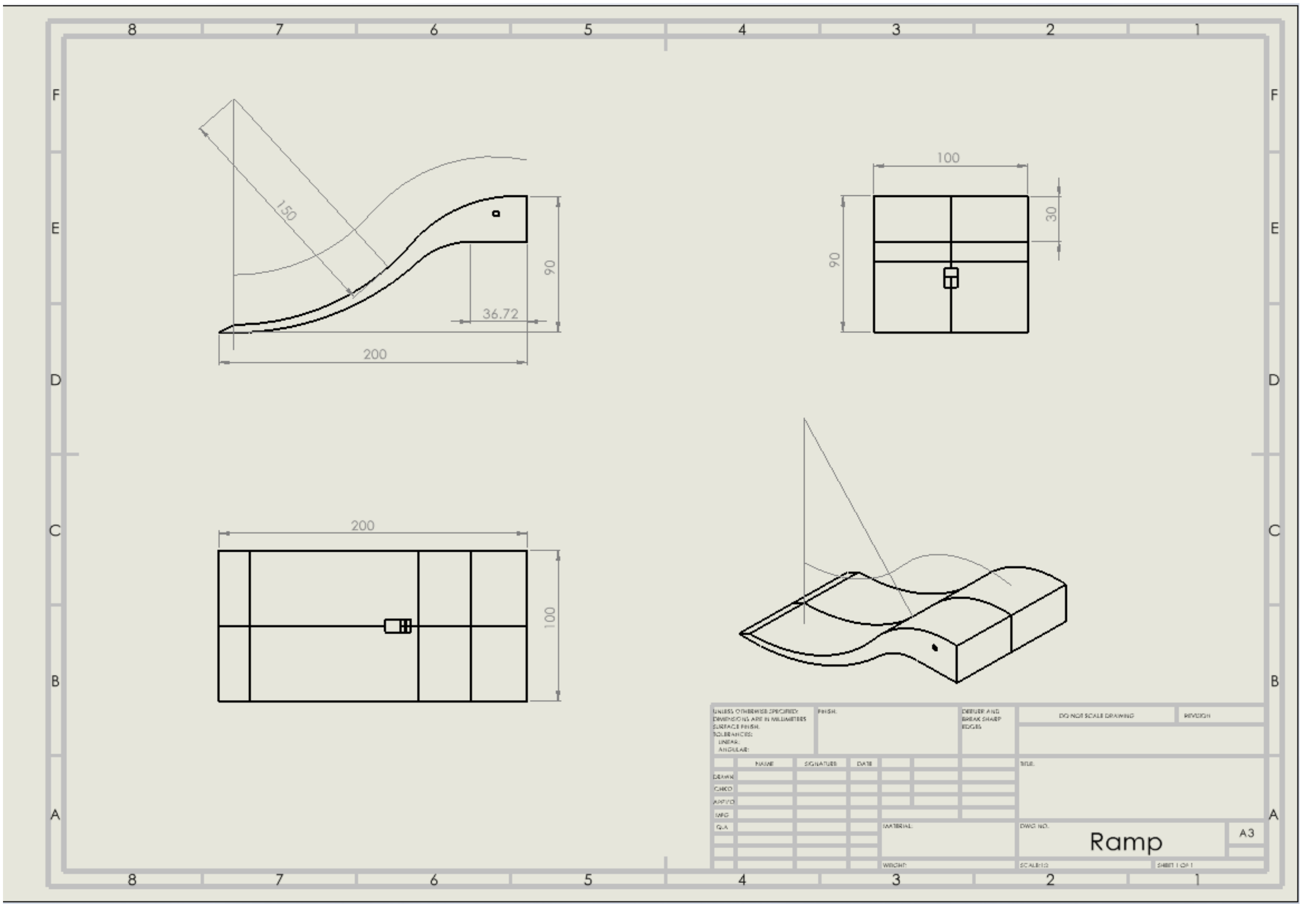


Figure 25: Ramp Drawing

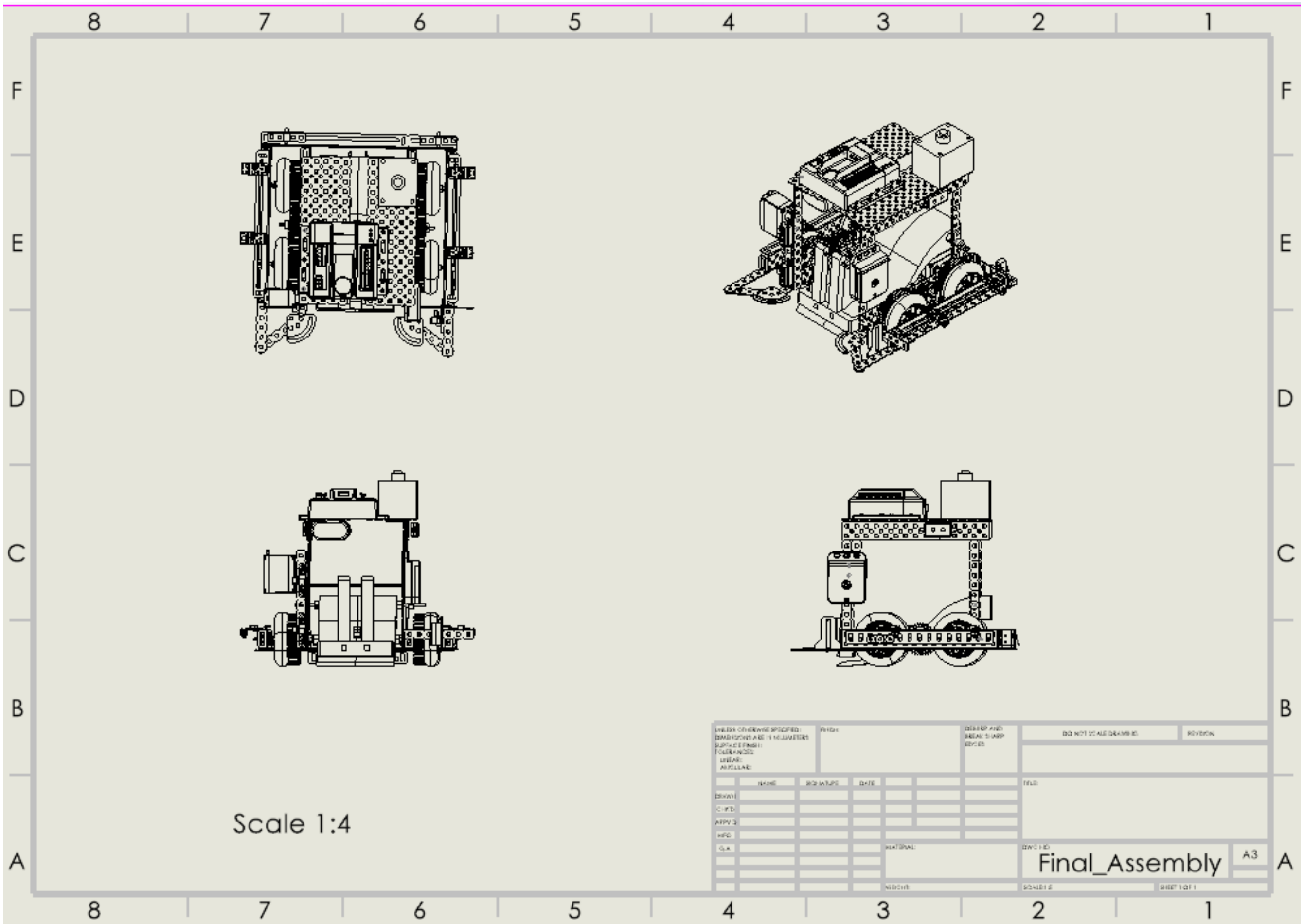


Figure 26: Final Assembly Drawing

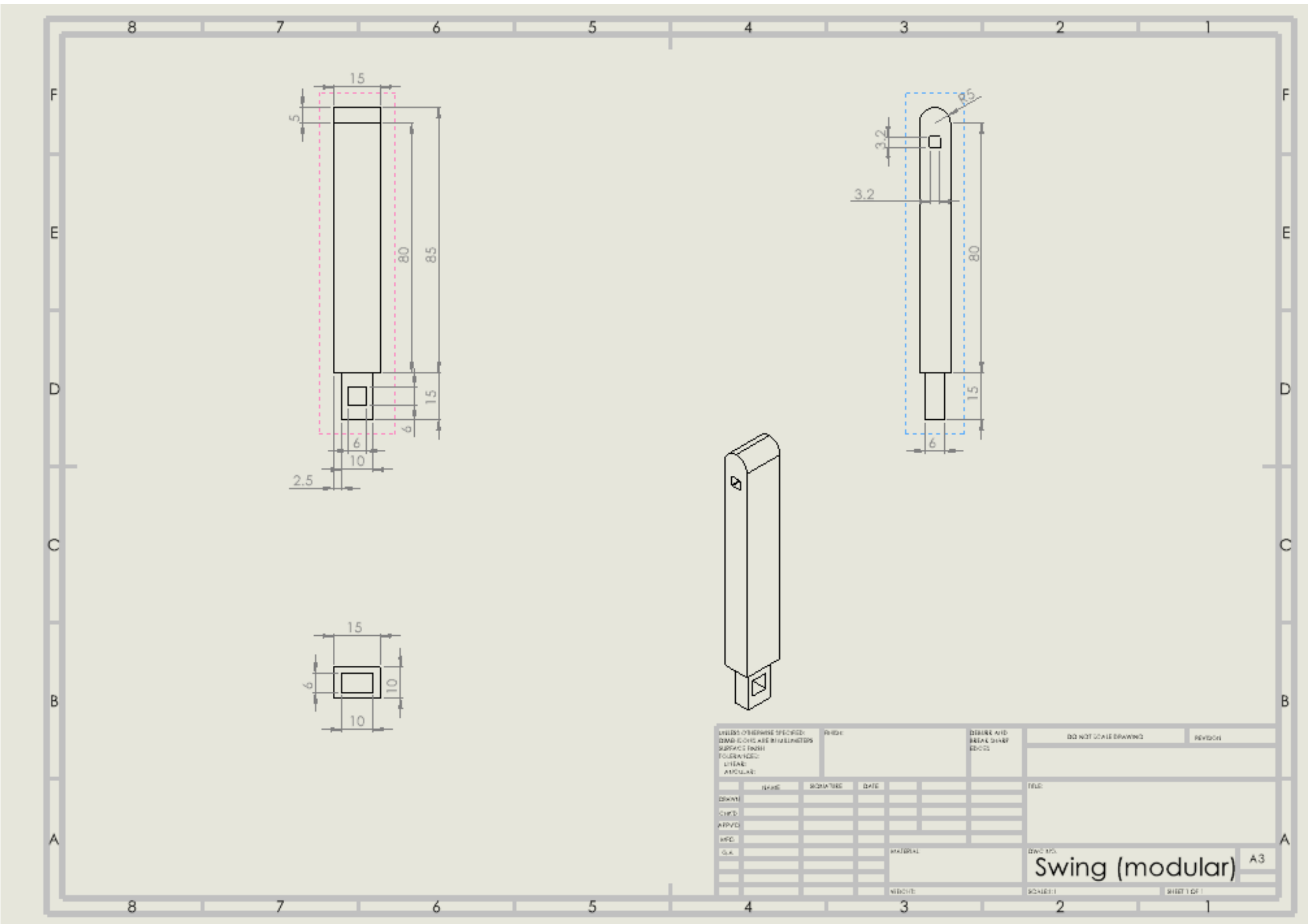


Figure 26: Swing Drawing

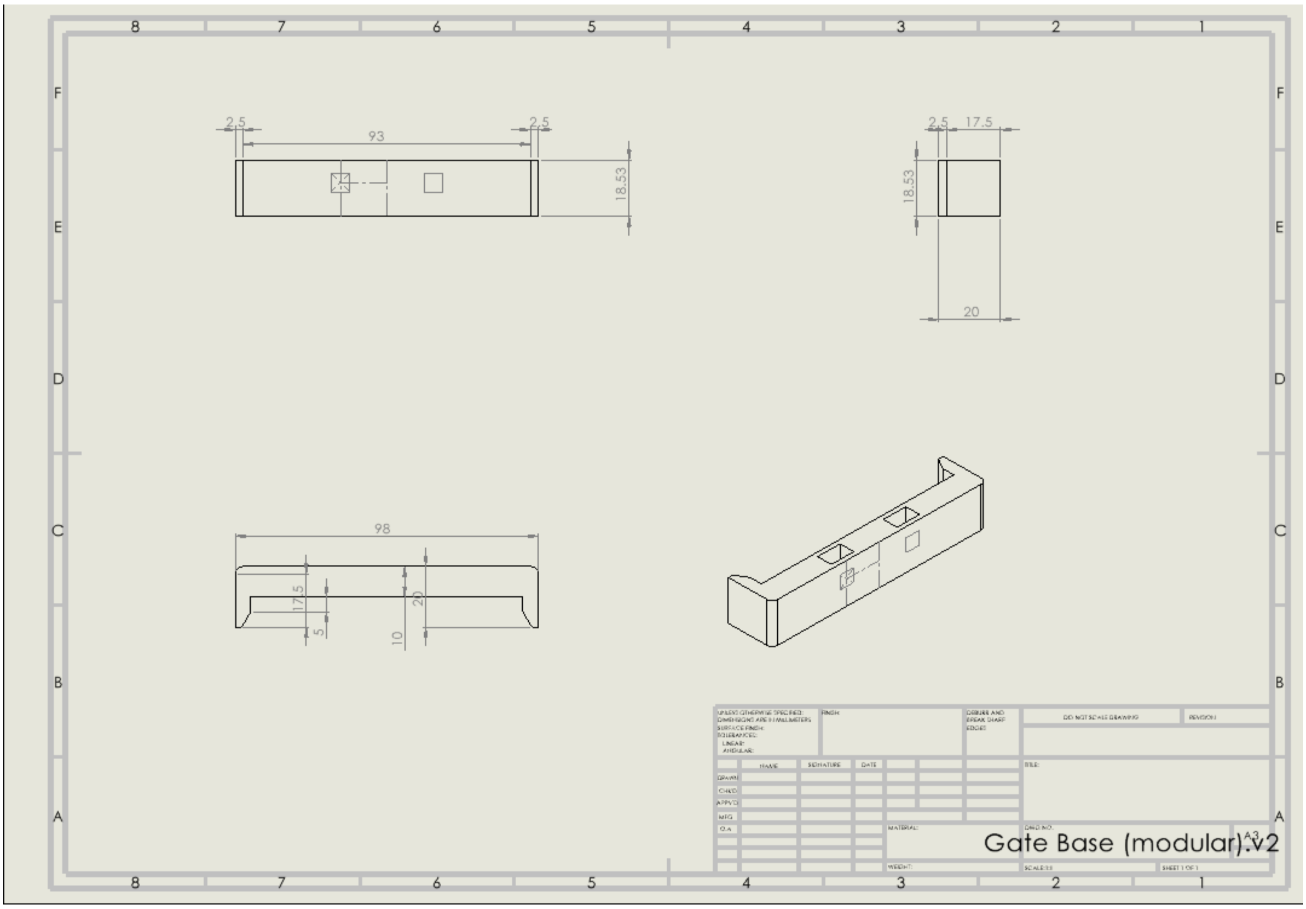


Figure 27: Gate Base Drawing

6. Control System

6.1 Sensors, Actuators, and Microprocessors

6.1.1 Sensors

Sharp Distance Sensor

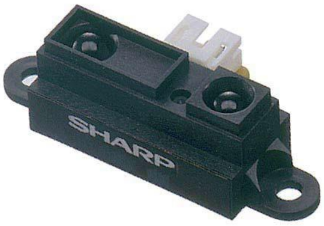


Figure 29: Sharp Distance Sensor.

Sharp Distance Sensor is a distance-measuring sensor unit, composed of an integrated combination of Position Sensitive Detector (PSD), Infrared Emitting Diode (IRED), and Signal Processing Circuit. This device outputs the voltage corresponding to the detection distance, as shown in Figure 30.

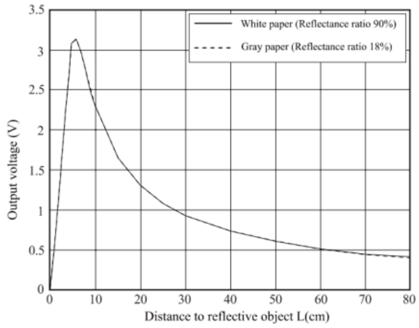


Figure 30: Sharp Distance Sensor (10-80cm) Distance V.S. Output Voltage.

Note that the output voltage is not completely proportional to distance. However, at the range of approximately 10 cm to 80 cm, the output voltage is inversely proportional to distance. Therefore, the effective detection range of this Sharp Distance Sensor is between 10 and 80 (cm).

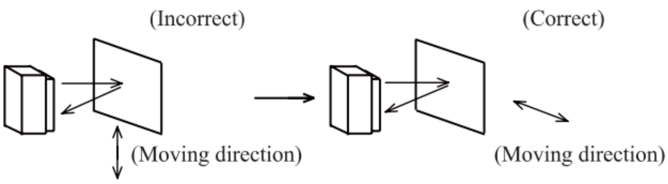


Figure 31: Suggested Orientation of Sharp Distance Sensor.

In order to decrease the deviation of measuring distance by moving direction of the reflective object, it shall be suggested to the sensor that the moving direction of the object and the line between the emitter centre and detector centre are vertical.

IR Line Tracking Module

Figure 32: Infrared Line Tracking Module.

The Infrared Line Tracking Module utilises a TCRT5000 Reflective Sensor, which includes an Infrared Emitter and Phototransistor in a leaded package enclosed by a Signal Processing Circuit.

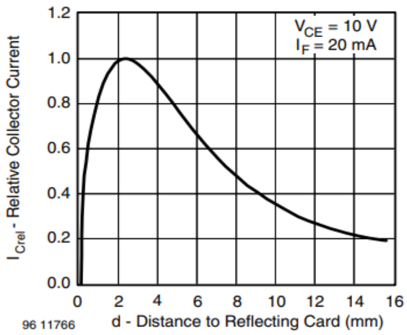


Figure 33: Distance to Reflection Card V.S. Relative Collector Current.

Note that, Figure 33 shows at the range of approximately 2 to 16 (mm), it is expected to see the inversely proportional relationship between distance to reflective card and relative collector current. Therefore, it is recommended to install the Infrared Line Tracking Module between its effective detection range. Moreover, it is also suggested to use digital output for robust detection.

Digital Compass 1490

Figure 34: Digital Compass 1490 in a Signal Processing Circuit.

The Digital Compass 1490 will sink up 25 mils per output channel, and it will switch so that no more than two adjacent output channels are activated at any one time. The output of Digital Compass 1490 is shown in Figure 35.

Direction	LED's	Decimal	I/O Read
North	0001	1	15
North-East	0011	3	12
East	0010	2	13
South-East	0110	6	9
South	0100	4	11
South-West	1100	12	3
West	1000	8	7
North-West	1001	9	6

Figure 35: Digital Compass 1490 Output.

It should be operated in a vertical position as the sensor indicates the horizontal component or compass component of the earth's field. If off-vertical, some of the vertical components of the earth's field are introduced which may create some directional error. Generally, tilt up to 12 degrees is an acceptable error. Other than installing it vertically into the signal processing circuit, it is strongly recommended to put Digital Compass 1490 away from any device with electrical current flows, to avoid unnecessary errors caused by induction.

Vex Shaft Encoder

Figure 36: Vex Shaft Encoders

Vex Shaft Encoder is a Quadrature type of encoder. It has a resolution of 360 counts per revolution with 2 count intervals and can distinguish between clockwise and counterclockwise rotation. Two output channels are required to transmit its sensor data to the Vex. It is suggested to connect Vex Shaft Encoder to Vex motors that need accurate positional control.

Limit Switch

Figure 37: Limit Switch

Limit Switch is a passive sensor that reads signals upon being pressed. It has 2 states as shown in Figure 38:.

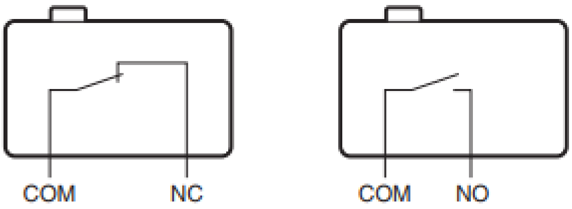


Figure 38: Normally Close (NC) & Normally Open (NO).

It is suggested to connect the Limit Switch with the Normally Open state, to detect if the vehicle touches the box.

6.1.2 Actuators

Vex Continuous Rotation Motor (276-2163)

Figure 39: Vex 3 Wires Continuous Rotation Motor 393

Vex 3 Wires Continuous Rotation Motor 393 is the primary actuator used in the VEX robotics design system. Build rotational mechanisms using this motor with a controller inside. This motor can either be configured into “high speed” and “turbo” versions by following the gear change procedure.

Vex 2 Wires Motor 393

Figure 40: Vex 2 Wire Motor 393

Similar to the aforementioned Vex 3 Wires Motor, Vex 2 Wire Motor 393 is a commonly used actuator in the VEX robotics design system. However, Vex 2 Wire Motor 393 does NOT have a built-in controller. Therefore, an external motor controller is needed to operate this motor.

6.1.3 Controllers

Arduino Mega

Figure 41: Arduino Mega 2560

Arduino Mega 2560 is an exemplary development board dedicated to building extensive applications. The board accommodates the ATmega2560 microcontroller, which operates at a frequency of 16 MHz. The board contains 54 digital input/output pins, 16 analog inputs, 4 UARTs, a USB connection, a power jack, an ICSP header, and a reset button. The recommended operating condition of Arduino Mega 2560 is shown in Figure 42.

Symbol	Description	Min	Typ	Max	Unit
V_{IN}	Input voltage from VIN pad / DC Jack	7	7.0	12	V
V_{USB}	Input voltage from USB connector	4.8	5.0	5.5	V
T_{OP}	Operating Temperature	-40	25	85	°C

Figure 42: Recommended Operating Condition for Arduino Mega 2560

Do note that the analog pins of Arduino Mega 2560 only provide 10 bits of resolution. By default, they measure from ground to 5 volts.

Vex-ARM Cortex

Figure 43: Vex Arm Cortex

Vex Cortex is a 32-bit ARM Cortex processor with wireless communication and code download using VEXnet. It features ten motor outputs that can deliver up to 8 A combined current, 12 digital input/outputs, 8 analog inputs, and three communication interfaces.

Do note that the analog pins of the Vex Arm Cortex are 32-bit resolution. By default, they measure from ground to 5 volts.

Motor Controller 29

Figure 44: Motor Controller 29

The Motor Controller 29 regulates the speed of a VEX motor based on a signal it receives from a VEX microcontroller. This allows for control of any VEX motor that does NOT have a built-in motor controller, such as VEX 2 Wire Motor 393.

6.2 Arduino V.S. VEX Control System

6.2.1 Hardware Comparison

The Arduino Mega 2560 is a more general-purpose microcontroller, while the VEX Cortex is specifically designed for robotics education. This means that the VEX Cortex has a number of features that are specifically designed for robotics applications, such as a built-in motor controller and a gyro sensor. However, the Arduino Mega 2560 has a larger number of input/output pins than the VEX Cortex. This means that the Arduino Mega 2560 can control more sensors and actuators than the VEX Cortex.

6.2.2 Original Controller Choice

The Arduino Mega 2560 was chosen as the first choice for the sensor system due to the team's familiarity with working with it previously and a relatively simple setup procedure. Examining the Arduino Mega 2560's wiring schematics shown in Figure 45, the sensors are placed on a breadboard with a sort of symmetry to represent the left and right sides of the vehicle. The sharp sensors are placed first for the consideration of the ball detection system while the line trackers are placed around and close to the back for detection of boundaries.

Programming the Arduino Mega 2560 was done through the Arduino Integrated Development Environment (IDE) where separate programs were written to test out the sharp sensor thresholds, the limit switch, and the line tracking module.

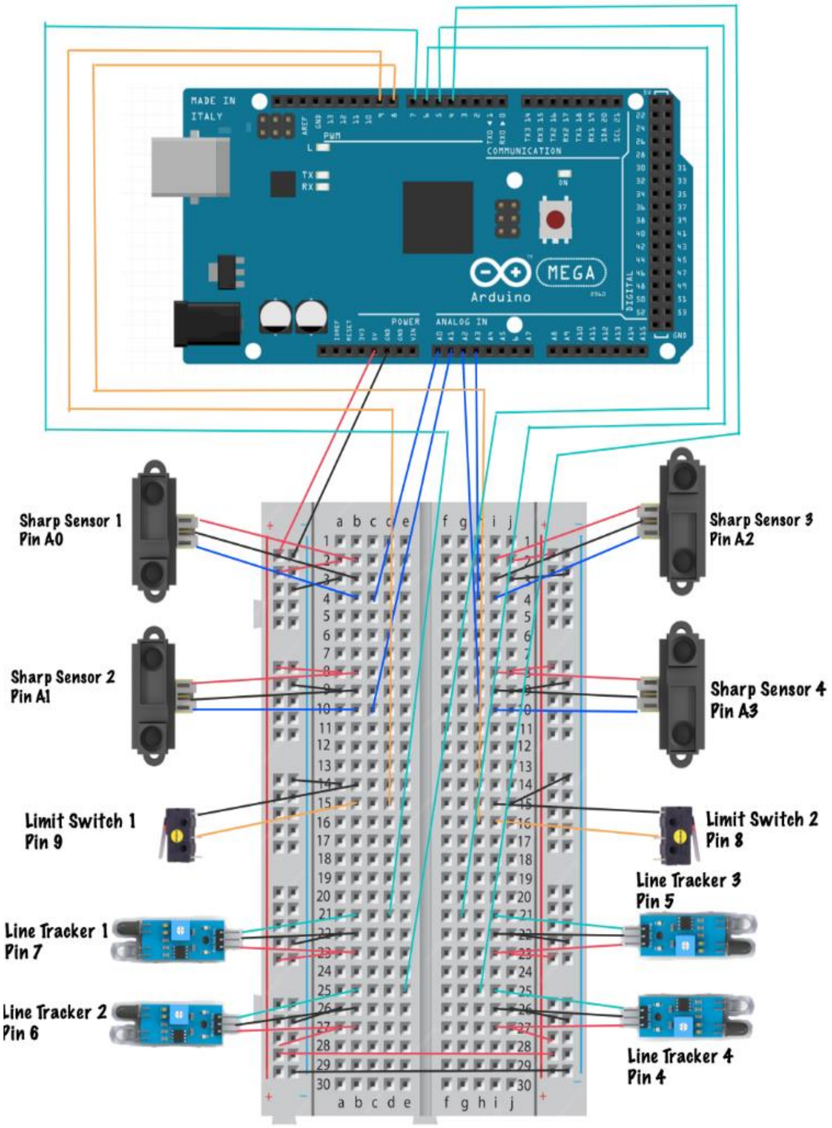


Figure 45: Wiring Diagram Using Arduino Mega 2560

6.2.3 Problems raised when using Arduino Microcontroller

Problems faced with the Arduino Mega 2560 included the need to communicate with the VEX microcontroller in order to send sensor signals to trigger commands to the VEX which controls the motor systems. Upon building up UART communication between Arduino and VEX ARM Cortex, the actual communication was not as smooth as the team intended it would be because of synchronisation issues.

From an electrical perspective, there were issues in figuring out how to power up both the Arduino and the VEX microcontroller as the battery given was meant for the VEX system only.

In terms of the mechanical assembly, the breadboard and the Arduino take up too much space on the top of the frame where the team has decided to place the control system at. Orientation to fit all three control components was difficult to figure out and the wiring had to be constantly extended as it was no longer a symmetrical design.

Most importantly, the signal resolution of the Arduino Mega 2560 was a significant issue. Arduino Mega 2560 analog pin only provides 10-bit resolution, however, VEX ARM Cortex provides 32-bit analog resolution. A quick comparison can be found in Table 26.

Table 26: Resolution Comparison Between 2 Microcontrollers.

Microcontroller	Resolution	Voltage	Converted Resolution
Arduino Mega 2560	10-bit	0 - 5 (Volts)	4.883 (mili-Volts)
VEX ARM Cortex	32-bit	0 - 5 (Volts)	**1.164 (nano-Volts**)

As shown in Table.X, it becomes clear that choosing VEX ARM Cortex as the microcontroller is more beneficial as it results in a control system that is more detailed, sensitive and provides accurate analog sensor readings.

6.2.4 Final Decision

Considering possible power loss, communication issues, and mechanical difficulties, it was then decided to port over the Arduino Sensor systems set up completely over to the VEX ARM Cortex.

Setting up the VEX microcontroller was vastly different from the Arduino due to the pins and ports having a limited number of analog and digital pins available. The need for digital pins for the digital compass was also a consideration to read the orientation of the vehicle. These issues were resolved by adjusting the positions of sensors, reducing the use of a limit switch and discovering that the number of analog pins and digital pins was just right for all our required sensors and motor controls.

Programming the VEX ARM Cortex was different from Arduino as it programs on a standard C programming language with its own keywords required in the ROBOTC IDE. Whereas Arduino requires programs in a C++ language. Though the two languages are similar, the syntax and variable types work differently hence another challenge was to rewrite the previous program written for Arduino so that it can be used for the VEX ARM Cortex.

The following Figure 46 illustrates the finalised wiring for the sensors and motors using the VEX ARM Cortex.

VEX ARM Cortex Wiring Diagram

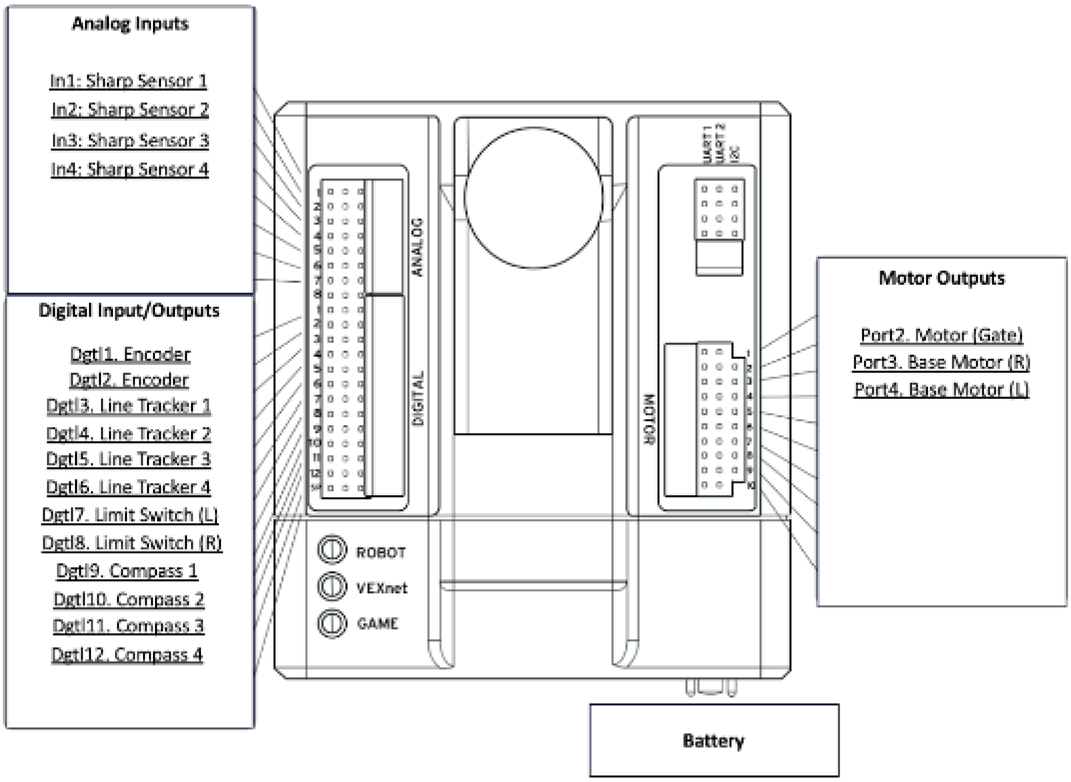


Figure 46: VEX ARM Cortex Wiring Diagram

6.3 Motors

6.3.1 Selection and Configuration

For this project, 4 Vex Motors are provided, as shown in Table 27 below.

Table 27: VEX Motor Specifications.

Quantity	Type	Stall Torque (Nm)	Free Speed (rpm)
2	393 (2-wire)	1.68	100
2	276-2163 (3-wire)	0.73	100

Based on the application of VEX motor in this project, it can be categorised into 2 functions: Gate Control Motor & Motion Control Motor.

Gate Control Motor responsible for 3 actions, namely open, close, and push. All these 3 actions are related to tennis ball collection. Figure 47 illustrates this well.

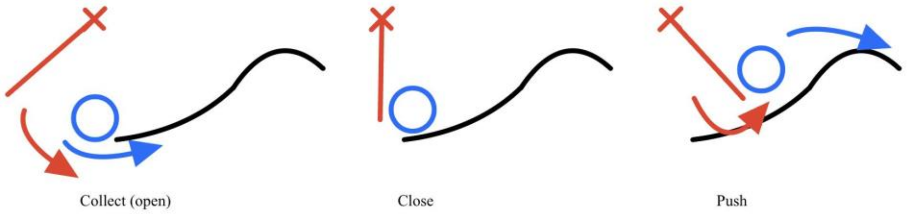


Figure 45: Schematic Illustration of Gate Control System

Motion Control Motor is comprehensive. As long as each base motor can rotate clockwise and counterclockwise, we can use Differential Drive to control the vehicle. Figure 48 shows differential drive control.

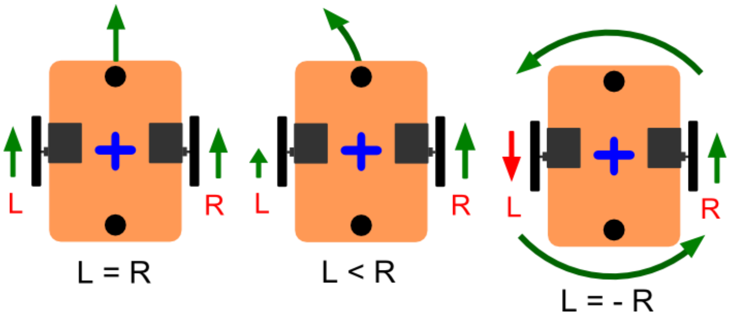


Figure 48: Differential Drive Vehicle Control

Main difference between the aforementioned 2 functions is whether or not the application case requires a high torque output. For the gate control motor, it needs higher torque output because it needs to hold its self-weight, collect the ball, and push the ball up. However, for the vehicle control motor, it does not need a high torque in operation.

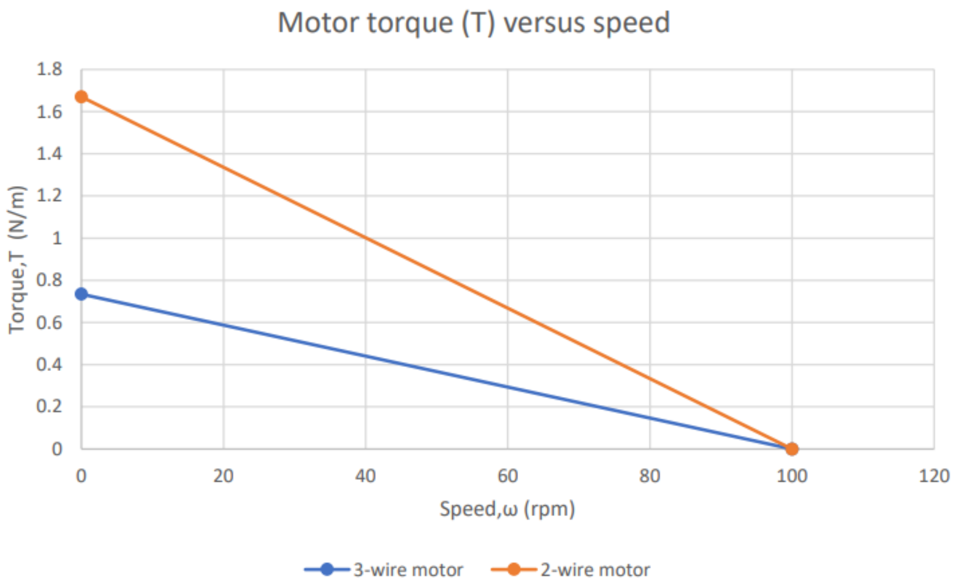


Figure 49: Visualization of Motor Specifications

As it is clearly shown in Figure 49, 2 wire motors have higher torque output regardless of rotation speed. Therefore, we may select 2-wire motor as gate control motor, and 3-wire motor as vehicle control motor.

6.3.2 Calibration

Vehicle Control Motor setup is simple, refer to the mechanical section of the report. However, for Gate Control Motor, it requires accurate positional control, thus an encoder needs to be set up together with a 2-wire motor.

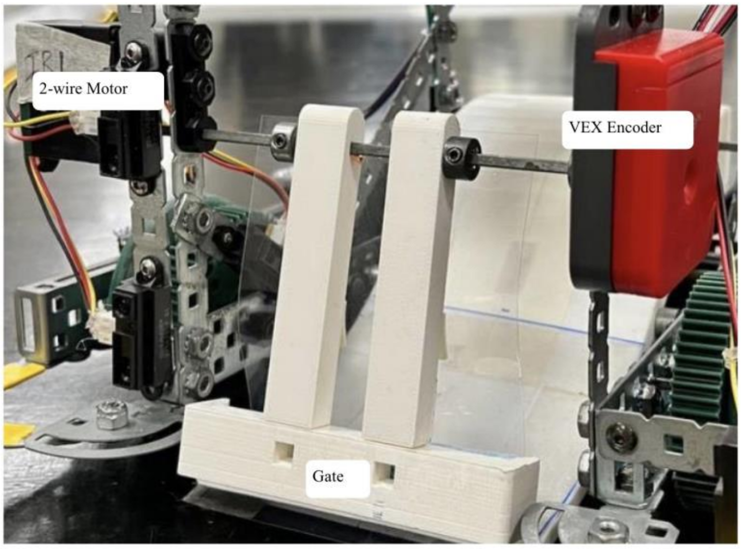


Figure 50: Motor Positional Control (Calibration) through VEX Encoder.

6.4 Sharp Distance Sensors

6.4.1 Selection & Configuration

For this project, 4 Sharp Distance Sensors are provided, as shown in Table 28: below.

Table 28: Sharp Distance Sensor Specifications

Quantity	Type	Effective Measuring Distance (cm)
3	GP2Y0A21SK0F	10 - 80
1	GP2Y0A41SK0F	4 - 30

Based on how the Sharp Distance Sensors are utilised, it can be categorised into 2 functions: Searching & Checking.

Searching

Searching refers to using Sharp Distance Sensors to search in the field. It requires a configuration that, using a certain combination of Sharp Distance Sensors, the vehicle is able to distinguish tennis balls and an opponent. The conceptual design of the searching mechanism is shown in Fig.X below.

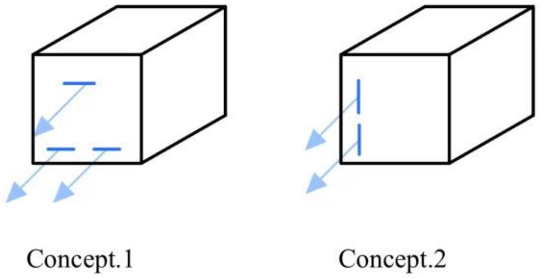


Figure 51: Conceptual Searching Strategies

Both concepts use the height difference to distinguish tennis balls and opponents. The lower distance sensor should be located at the same level as the centre of the tennis ball, and the higher distance sensor should be located at least above the top of every tennis ball. Such a configuration can achieve the task of distinguishing the ball and opponent as illustrated in Figure 52. Ball detected if the higher sensor reads False but lower sensor reads True, and opponent detected if both sensors reads True.

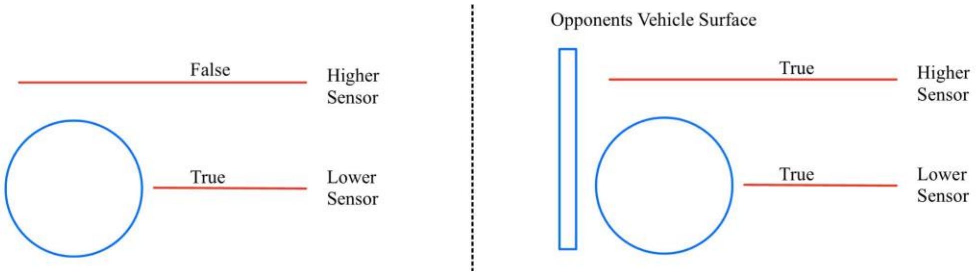


Figure 52: Using Hight Difference of 2 Sharp Distance Sensor to Distinguish Objects

Concept 1 places two sharp distance sensors at the lower level to increase the ball searching line (2 straight parallel arrows) to two parallel lines. Concept 2 places only one sharp distance sensor at the lower level, therefore, it only has one ball searching line (one straight arrow). Although Concept 1 has a relatively higher possibility of finding a ball, Concept 2 was adopted at the end. This is because our search method involves rotating the vehicle, making a single line sufficient to sweep the entire rotation for ball detection. Consequently, one of the lower-level sensors in Concept 1 becomes redundant in this scenario.

To guarantee a larger effective searching range, 10 - 80 cm Sharp Distance Sensors are selected for both higher-level opponent detection sensor and lower-level ball detection sensor.

Checking

After searching, the next step is to check when to collect the ball, and check if the ball is successfully captured. To do so, a checking mechanism is needed as shown in Fig.X.

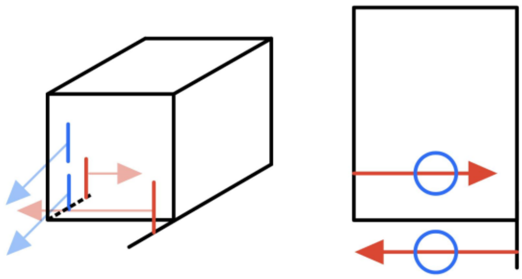


Figure 53: Isometric and Top View of Checking Mechanism

The checking mechanism requires 2 Sharp Distance Sensors. One outside the gate to check if a tennis ball is within the collectible area. The other one is inside to check if the ball is successfully captured and secured.

The inner sensor is only responsible for checking if a tennis ball is secured within the vehicle. A Sharp Distance Sensor with a 4 -30 cm range is sufficient for this purpose. The outer sensor needs to check if a ball is present in the collectible area regardless of whether the ball is moving or not. Thus, a 10 - 80 cm Sharp Distance Sensor is chosen.

6.4.2 Orientation

In the previous section, it was mentioned that it is suggested to place the Sharp Distance Sensor Vertically rather than horizontally. Figure 52 shows exactly why horizontal orientation is not preferred.

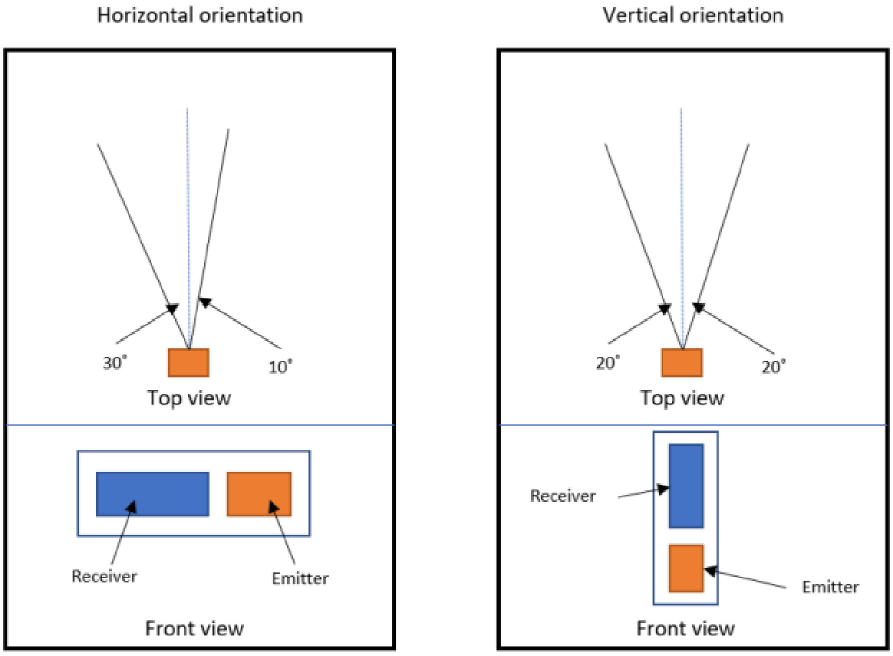


Figure 54: Effective Detection Angle and Sharp Distance Sensor Orientation

Horizontal orientation will yield an asymmetric effective detection angle and such asymmetricity is not preferred because it effectively reduces the ball search area. Therefore, a vertical orientation is much more preferred in this application.

The image below shows the final sharp sensor placements on the vehicle.

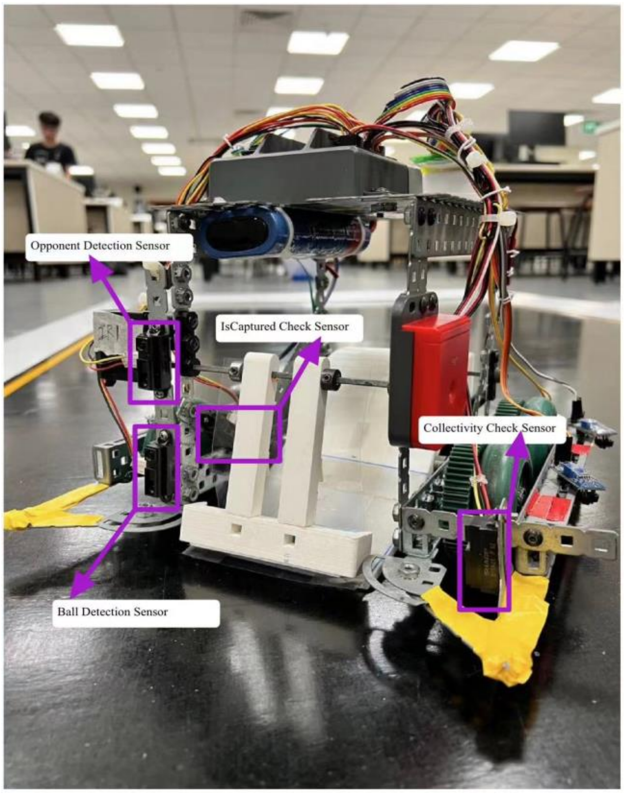


Figure 55: Actual Sharp Distance Sensor Configuration

6.4.3 Calibration

During the calibration process of the sharp sensor, it was discovered that the sensor's accuracy varied depending on its resolution. The sensor's accuracy was found to be inversely related to its resolution, meaning that the higher the resolution, the lower the accuracy. Using the Arduino Mega 2560 previously, the sharp sensor recorded different readings compared to using the VEX ARM Cortex due to the fact that the VEX microcontroller has a higher resolution of 32-bits. The VEX microcontroller was more accurate and sensitive to changes of position hence new thresholds were re-tested when the sensors were ported from the Arduino to VEX microcontroller

The first method of ball detection the team tried was based on the time elapsed of the scanning process which was later changed to direct detection of the threshold value due to the complications of using a timer and overall ineffectiveness of the search strategy. With further testing of direct ball detection functions, it was observed that the threshold values were fluctuating and thus making ball detection difficult, thus calibration was required.

Trying out different positions and ranges the sensor can detect, the fluctuation still continued where values would show two different extremes where one would be the required reading but would jump to another promptly after displaying the first reading. This was later discovered to be due to the sharp sensor sensing within a cone-shaped radius as shown in Figure 56.

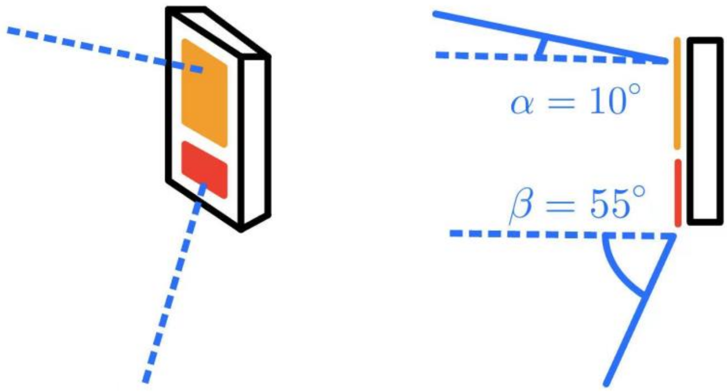


Figure 56: Sharp Distance Sensor Detection Deviations

Attempts to overcome this cone shape detection radius were to cover up one part of the sensor which ended up causing the threshold value to be inaccurate and less sensitive to detection changes. To overcome this issue, the physical position of the sensor was adjusted from horizontal to vertical. The sensor's accuracy was improved because the sensor's field of view was affected by its position, where a vertical position would force the detection range to be parallel to the vehicle frame and not in widespread angle anymore. The sharp sensor could then be made to focus more accurately on the object being measured and thus allowing the ball detection function to perform as intended.

6.5 Line Tracking Module

6.5.1 Configuration

For this project, four TCRT5000 IR Line Tracking Sensors are provided. As this is the only IR Line Tracking Sensor Provided, there are no other options to choose from. Therefore, the team proceeded to IR Line Tracking Sensor Configuration.

The original line track module configuration is shown in Figure 57. This design is straightforward by putting a line track sensor at each corner of the vehicle. However, such configuration may lead to failure of ball collection if the ball is near the boundary. Since our ball collection mechanism uses a swinging gate that extends out beyond the base when collecting the ball, the ball needs to touch the vehicle base before the gate closes. In this case, if the ball is on the boundary yellow line as indicated in the second diagram of Figure 57, such configuration may cause the line tracking module to activate before the ball is collected.

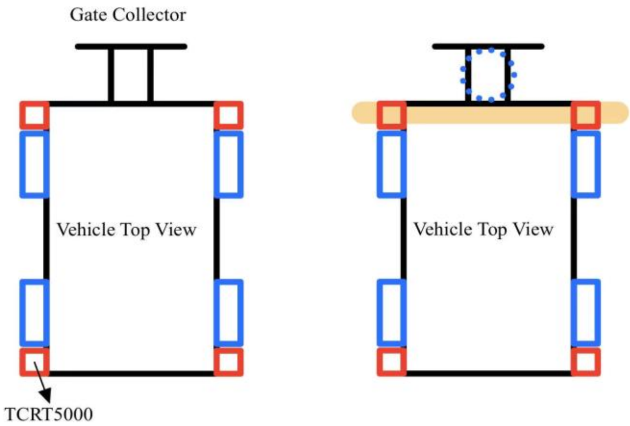


Figure 57: Original IR Line Tracking Module Configuration and its Disadvantage

Given the fact that the vehicle can touch the yellow boundary (i.e. it will not be considered as out of the field as long as the vehicle does cross the yellow line to the ground outside), the team took this into consideration to re-configure the line tracking module, in order to achieve greater flexibility in searching and collection. Therefore, the final IR Line Tracking Module configuration is shown in Figure 58.

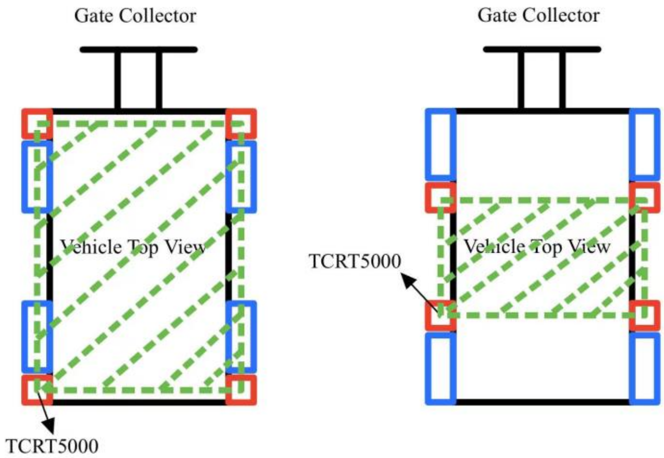


Figure 58: Finalized IR Line Tracking Module Configuration (On Right)

Figure 58 also shows the “Collision Box " of the boundary avoidance system. It is obvious that the final line track module configuration on the right has a relatively smaller “Collision Box”. Such a configuration gives the vehicle flexibility to explore across the yellow boundary, but at the same time guarantees it does not cross the arena boundaries. With this final configuration, the vehicle no longer has the issue that it can not collect the ball on the boundary.

6.5.2 Calibration

Configuring the Line Tracking module was done by taking digital readings of the line tracking module where a 0 would represent the initial reading where the module is not activated and a 1 would indicate a reflective surface being detected. The first challenge faced during line tracking configuration was the placement angle at which the module would easily be activated when the boundary line is being detected. This was decided to be placed close to the wheels and specific angles for each different sensor after testing out when the sensor would first activate within the competition field. Eventually, all line tracking sensors are calibrated to their ideal position and orientation. Figure 59 shows the collision box of the actual line tracking system on the vehicle. Although it seems to be small and dangerous (i.e. dangerous means potentially our vehicle may cross the arena boundary), rigorous testing was done to ensure that this is the optimal configuration.

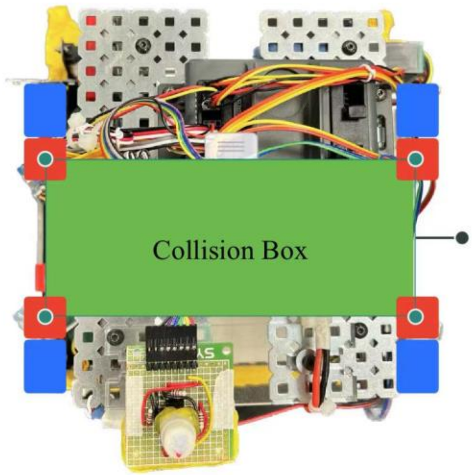


Figure 59: Collision Box of the Line Track System

The second challenge in configuring the Line tracking module was running the module parallel with the main program. The line tracking module function had to send commands to the motor whenever the boundary was detected. This would clash with the other motor commands such as ball search or ball collection as multiple conflicting signals are given to the motor at once. The problem observed was that the vehicle prioritised one command before quickly executing the clashing input, causing erratic and unpredictable control of the vehicle. The line tracking module had to also be deactivated when returning the ball to the start position in order to deposit the ball as the delivery area had yellow boundary lines near the ball drop-off area.

The problem was resolved by researching and exploring the capabilities of the keywords to VEX microcontroller programming where different tasks were found to be able to run parallel at once under a main task. The line tracking module was given a standalone task which would be started and prioritised to control the motor should a boundary be detected and could be easily stopped with stop Task() function when the vehicle was returning to deposit the ball.

More programming logic was tested with the line tracking task to ensure different scenarios and situations can be overcome which is explained further in the Programming chapter.

6.6 Digital Compass

6.61. Configuration

For this project, a Digital Compass 1490 that senses the magnetic field of the earth to give directional output is provided. However, this also means that Digital Compass 1490 can be easily affected by the induced magnetic field of current. Therefore, it should be placed at a location that is away from any potential magnetic field interference. Based on this finding, it was placed at the farthest corner of the vehicle and we used a casing to isolate the compass as shown in Figure 60.

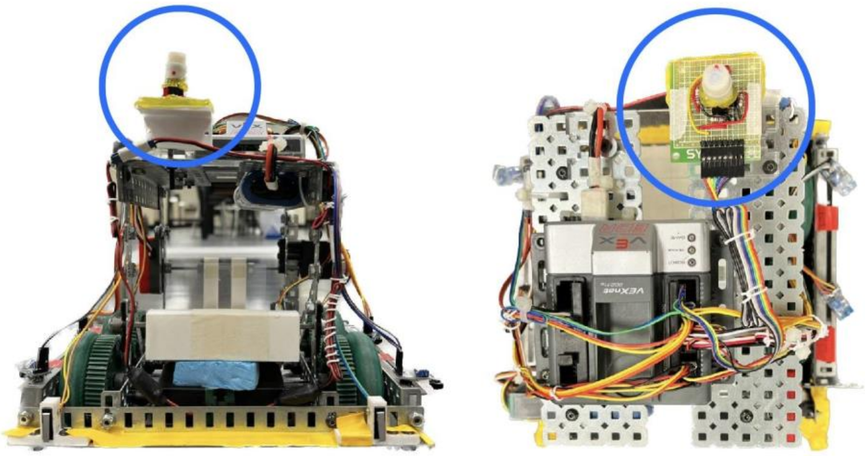


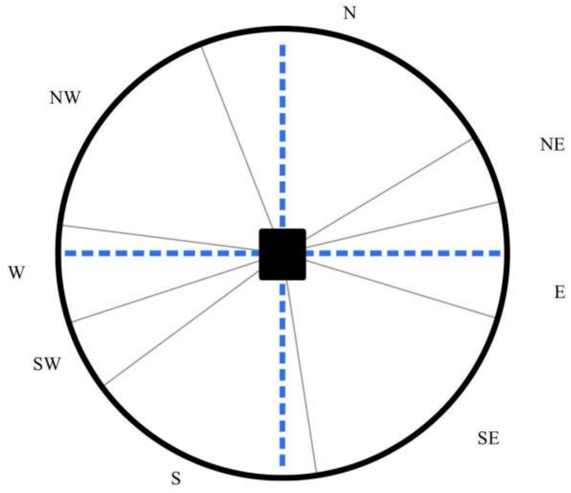
Figure 60: Digital Compass Placement and Casing Isolation

6.6.2 Calibration

Calibration of the digital compass was a major challenge due to the inaccuracy of the digital compass itself. The digital compass was a key component of the vehicle's performance as it was used for three major functions. Firstly, the digital compass was required for the vehicle to return back to the starting position in order to deposit the ball. For this task, the vehicle had to be oriented in its starting position (front-facing) as the ball deposition mechanism was at the back.

The digital compass was also key for the search strategy for balls where the vehicle would reorientate and reverse while doing a rotation search in the case where no ball was being collected and the vehicle had completed searching in the forward length of the field. It was also used in the line tracking module activation where the vehicle had to reorient itself back to searching in the case where the vehicle was near the boundary.

When testing the digital compass, it is found that there are specific ranges where the orientation would be picked up based on the digital readings of the compass. This range caused the digital compass to indicate a defined position has been reached within the software but physically the orientation was deviated. There were different orientation requirements for the three major functions. The most errors were encountered when attempting to reorient the vehicle for delivering the ball. For example, the program would show that the vehicle is facing the front but the vehicle is actually deviated to the left. This led to vehicles reversing out of the field instead of returning in the correct orientation to deposit the ball.



Direction	Angle
N	339 ~ 62
NE	62 ~79
E	79 ~ 111
SE	111 ~ 174
S	174 ~ 237
SW	237 ~ 253
W	253 ~ 274
NW	274 ~ 339

Figure 61: Experimental Digital Compass 1490 Directional Output

After testing, the team obtained the approximate output range of each directional output of Digital Compass 1490, as shown in Figure 61. The suggested reasoning for the inaccurate readings is that the compass was affected by the presence of the magnetic field which results in noisy sensor reading which ultimately affects its accuracy. Attempts to overcome this issue was to install an electrically insulating material to which the compass would be mounted but this did not overcome the issue despite rotating and repositioning the compass around the top of the vehicle.

As these errors are introduced by magnetic field interference, changing the orientation will not eliminate these errors.

The next solution attempted was to try out different digital compasses available for the same model and pick out the one with the most narrow range of the required orientation. However, once mounted on the vehicle, the readings continue to deviate from the actual physical orientation.

Finally, it was realised that the hardware calibration for this Digital Compass is not possible, therefore we resorted to programmatically resolving the issue. A program was written to try to compensate for the orientation by rotating the vehicle more, but this ended up getting stuck in a loop as the orientation was constantly reading the same value which kept the vehicle rotating. Eventually, this was overcome as we did an external reorientation by using the line tracking module instead.

We had originally stopped the line tracking module in order to avoid the case where the ball would not deposit when detecting the boundary line at the home position. However, we observed that the left deviation would always cause the back left line tracking sensor to activate. We allowed for the back left line tracking module to purposefully activate to move the vehicle forward and rotate straight again before turning off the line tracking task when returning to the home position to deposit the ball.

The other 2 major functions were less affected by the inaccuracy as the vehicle was programmed to stop before continuing other rotation actions which caused less impact.

6.7 Limit Switches

6.7.1 Limit Switch Configuration

Configuring the Limit switch was one of the more straightforward sensor configurations. Nonetheless, there were some challenges that were still faced. Our limit switch was connected via for a normally open logic. Two limit switches were used for the purpose of indicating that the vehicle had returned to the start position to deposit the ball as shown in Figure 62.

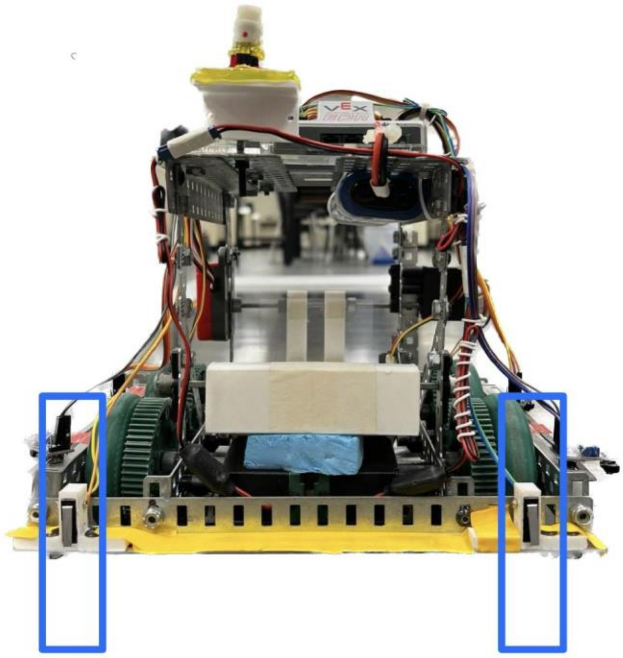


Figure 62: Limit Switch Configuration

The limit switch was placed at the back of the vehicle aligned to the wheels. The switches were placed vertically and secured with a 3D printed module to ensure firm attachment when withstanding the impact of the reversing vehicle. One issue was the ease of triggering the limit switch for which attempts to install a bumper were suggested. This was to ensure that the vehicle could easily deposit the ball even where there was a small distance as the limit switch can be easily triggered.

However, testing out the suggestion,thr vehicle faced the possibility of the ball not being deposited every time, hence the original position of the limit switches were more ideal despite the bumper being a more robust solution to triggering ball deposition.

7. Programming Logic

7.1 Finalised Hardware Structure

To better program the vehicle, it is very important to understand the hardware structure of the vehicle. As shown in Figure 63 below.

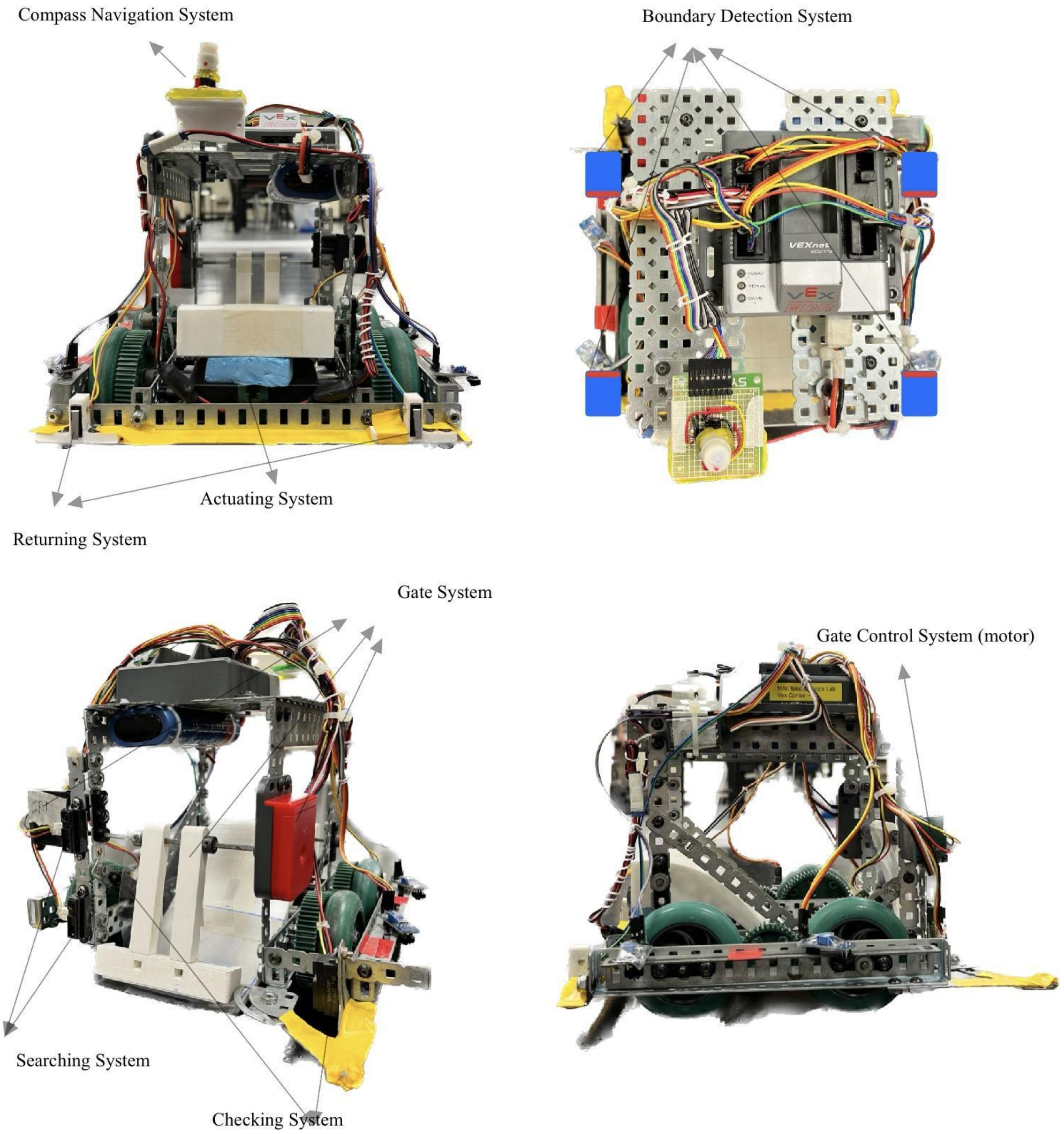


Figure 63: Modular System of the Vehicle

The vehicle can be decomposed into several systems:

1. Searching System
   It is responsible for searching, as well as distinguishing the tennis ball from the opponent. It serves as an eye for the vehicle to move within the arena.

2. Checking System
   Once the Searching system identifies the tennis ball, it is the Checking system's responsibility to check if the ball is within the collectible area and if the ball is successfully captured.

3. Gate and Gate Control System
   If the checking system indicates the ball is ready to collect, or the ball is successfully captured, then the Gate Control System will control the gate to collect the ball, secure the ball, and push the ball, respectively.

4. Boundary Detection System
   Boundary Detection System runs all the time except when the return state is activated (more illustration in this chapter later), to guarantee our vehicle explores and moves within the field without heading out.

5. Returning System
   The Returning System is to check if the vehicle is back to box and ready to drop off the ball. Only when checking system checked ball is successfully captured, then return system will be activated.

6. Compass Navigation System
   The Compass Navigation System makes use of digital compass to check if the vehicle is in a correct orientation. This is very important to both returning and searching system.

7. Actuating System
   The Actuating System is the most basic system of the vehicle. It supports all the aforementioned systems' motion. Except for the Gate Control System as it has its own motor.

7.2 Functions

Modular Functions

Based on the task performed by all aforementioned systems, we can categorize the tasks into several functions as shown in Table 29 below.

Table 29: Programmatical Overview.

Task	Return	Modular Function
Search, Collect, and Avoid	**void**	**searchWhateverInfront** ( )
Return to Box	**void**	**backToBox** ( )
Avoid Boundary	**void**	**isDetect** ( )

These functions are the high-level logic which does NOT interact directly with the hardware. Instead, the hardware interaction as well as commonly used functions are in utilities functions.

Important Utilities

void motorControl (int Index, int Speed, int Time)
Control the speed and operation time of specific motor.
Index: motor index
Speed: motor operation speed, range from -127 to 127
Time: time duration for [index] motor to rotate at [speed]

void gateControl (int Position)
Control the position of the gate motor.
Position: desired target position.

void gatePush ( )
Push up the gate at its maximum speed

bool isObejectInfront (int Index, int Threshold)
Filter, only 2 consecutive readings larger than the threshold is considered as object in front
Index: sharp distance sensor index
Threshold: when to consider a object detected

bool isCapturedt ( )
Check if ball captured successfully

bool isBall( )
Check if ball ready to collect

bool isReturn( )
Check if vehicle back to box already

bool readCompass ( )
Read compass direction

7.3 Vehicle Searching Strategy

Initial Strategy

According to the rule of the competition, all 3 tennis balls will initially be randomly placed at farther one half of the field. Regardless of our initial position is left or right, our vehicle needs to go forward one half one the field at its maximum speed.

The reasons to use its maximum speed is because, on the one hand, initially the nearer one half of the field has no ball inside, therefore we prefer not to waste time on it. On the other hand, our searching strategy requires the vehicle to spin. Therefore, it will be a good advantage if initially we can go farther than our opponent as illustrated in Figure 64.

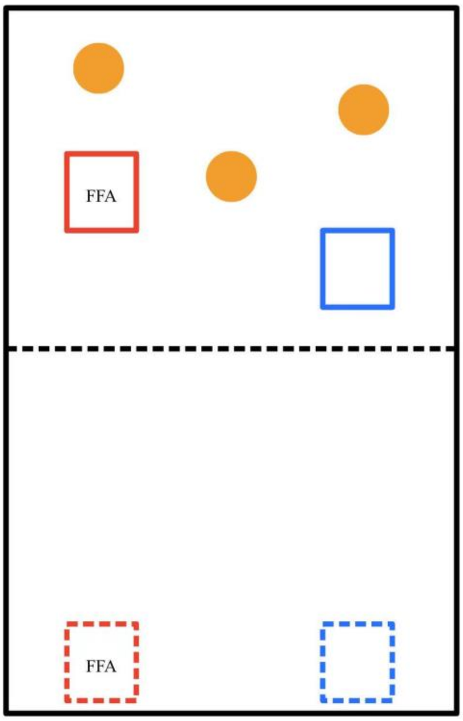


Figure 64: Speed advantage illustration

As shown in Figure 64, in such a scenario, if we are ahead of our opponent, there will be a higher possibility for our vehicle to spin and find the middle tennis ball, which is a huge advantage to us. Other than this, ahead of our opponent also means lesser chance for our searching system to mistakenly consider our opponent as the ball. Although we do equip with opponent detection function, sometimes this function may fail. More illustration will be discussed in searching strategy.

7.4 Searching Strategy

void searchWhateverInfront ( )

Recall the previously discussed hardware configuration of sharp distance sensors. Figure 65 is a brief visualization of how our searching mechanism and strategy works.

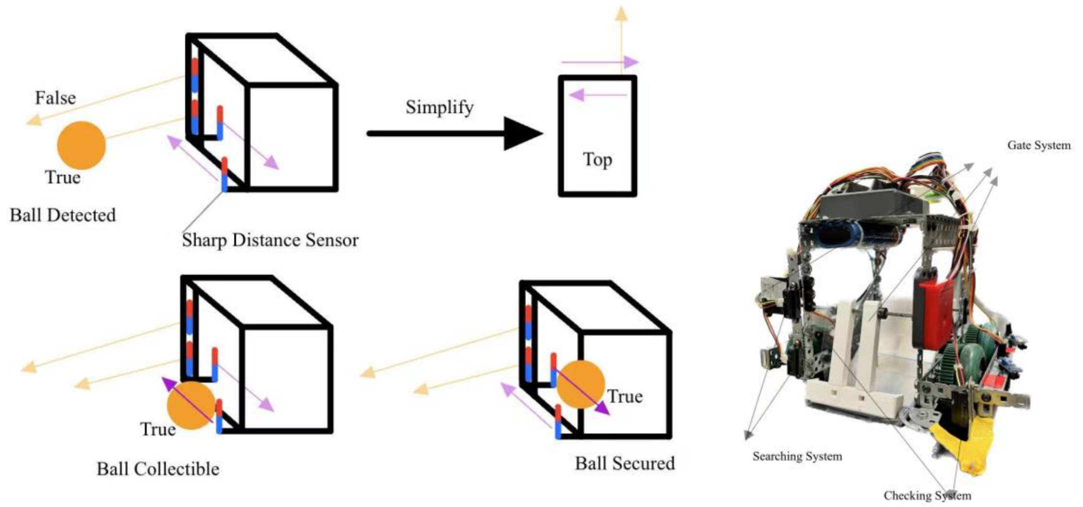


Figure 65: Visualization of ball searching and checking system

Our searching mechanism uses the height difference to distinguish a ball and an opponent as shown in Figure 65 diagram 1. For example, if the bottom sensor reads True and the top sensor reads False, that means a tennis ball is found. However, if both sensor reads True, that means an opponent is found.

Because our sensors are installed at the side of the gate, right after the tennis ball is detected, our vehicle will correct its orientation to guarantee it is facing the ball. After correction is done, our vehicle will move forward until the first checking sensor checks the ball is ready for collection. **Then the gate will close while our vehicle moving forward.** If the collection process is smooth, the ball will enter the vehicle and get detected by second checking sensor to check if the ball is successfully captured.

Limitation

However, our opponent detection mechanism does NOT work very well due to 2 reasons. First, when designing this search system, we assume our opponent is a solid structure. However, in real cases, most of our opponents are actually hollow structures, making the top sensor hard to detect. Second, the 2 sharp distance sensors will inevitably interfere with each other, this undesired interference will lower the detection accuracy for both sensors. Given the fact that our primary goal is to successfully locate, collect, and deliver a ball, we calibrate the bottom ball detection sensor as much as possible. But the top opponent detection sensor is what we used to calibrate with, in such a case, top sensor accuracy still affected by interference.

Program Pseudocode

```
void seachWhateverInfront()
Loop the task of finding and collect the ball
        If (ball find && opponent not found)
            Forward to check ball
            Ball checked then collect
            If success: break the loop
        Else if (opponet found)
            Avoid
            Correct direction
```

Program Flowchart

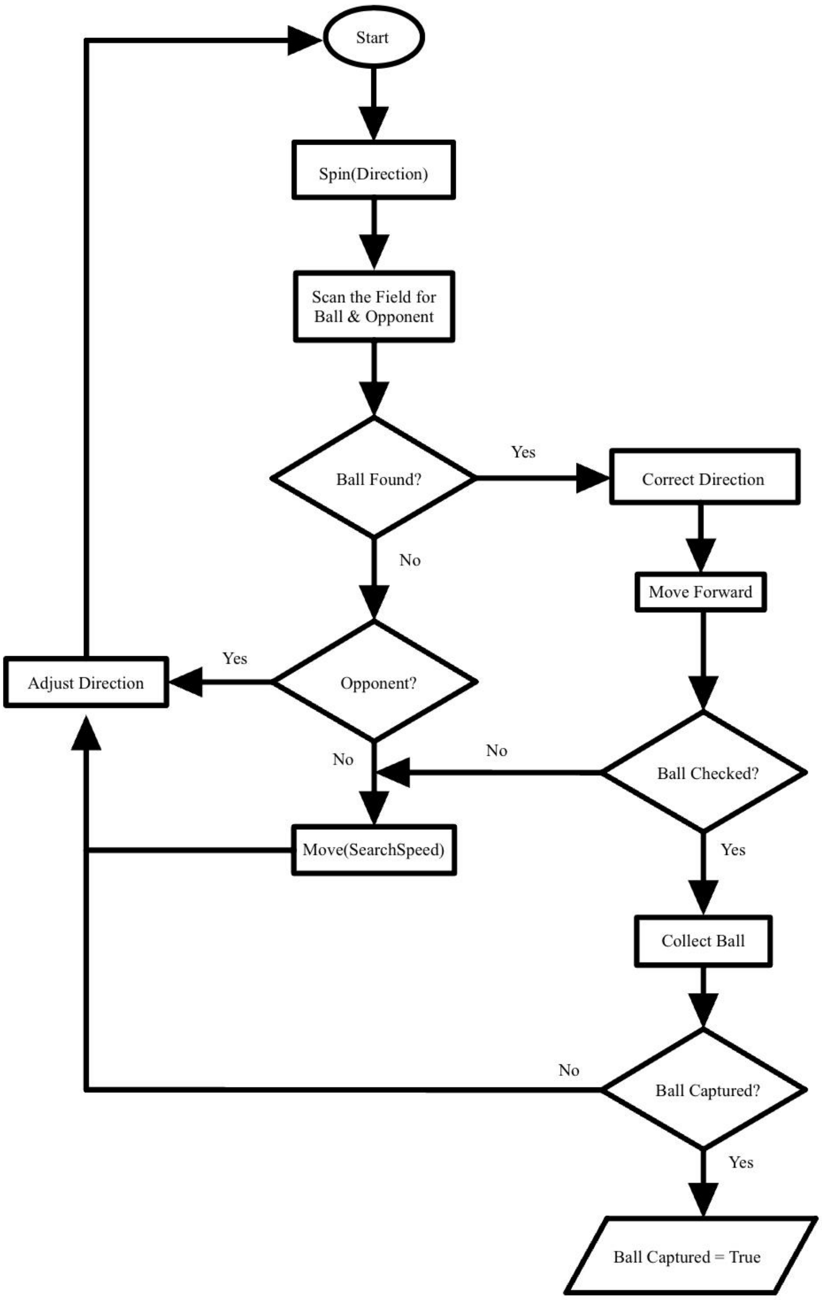


Figure 66: Searching strategy program flowchart

7.5 Vehicle Return Strategy

Visualization of Returning strategy

void backToBox( )

Recall the returning system, our vehicle has 2 limit switches at back. After checked ball is successfully captured (ballCaptured = True), then backToBox function will be activated. A visualization of this returning strategy is shown in Figure 67 below.

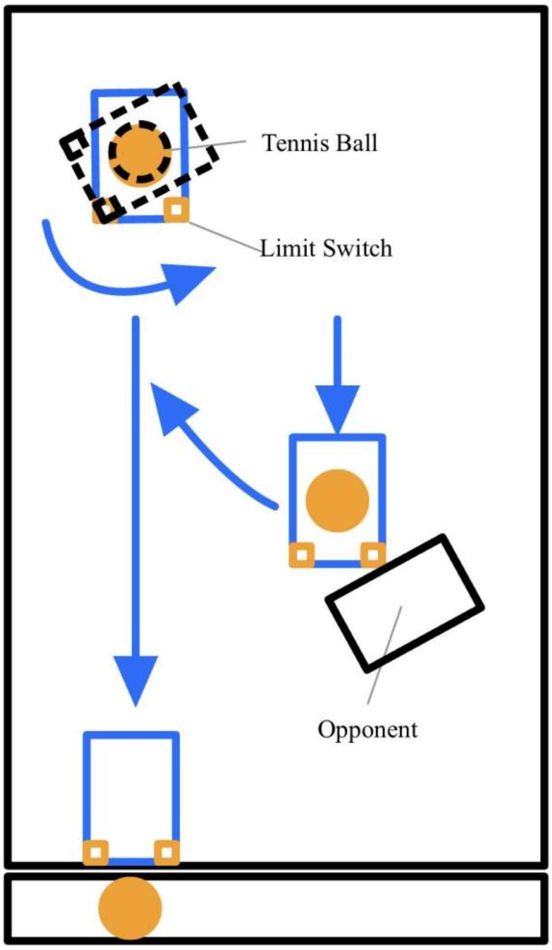


Figure 67: Visualization of Returning Function

Once the ball is successfully captured, regardless of the current orientation, our vehicle will correct its orientation based on the digital compass reading. Then it will start to go backward. During the back to box process, our vehicle might potentially hit an opponent or back with a deviated path due to the inaccuracy of the compass. If any of the case happens, our vehicle will correct its position and orientation, then continuous the task of delivering.

Limitations

The contact area of the limit switch is not big enough. Although we have tried to add a bumper structure, it will lead to other issue, even causing failure of ball detection. The elongated (bumper) structure acts as a lever, making the limit switch too easy to be triggered. We can NOT guarantee 2 limit switches are activitate at the same time. In such a case, no matter which side of the limit switch is activated first, the ball deliver will not be completed.

However, if without bumper struction, the force needs to activate the limit switch will be relatively higher. In this case, even one of the limit switch touches the box, as long as our vehicle is not hitting the box directly, it will not be activated. But as mentioned before, such a configuration is also quite hard for the opponent to activate it.

Program Pseudocode

```
void backToBox( )
    Stop the robot
    Loop untill compass give correct facing
    Loop until ball delivered
        If one limit switch activated: means opponent, avoid
        If line tracking sensor activated: means path tilted, correct it
        If both limit switch activated: reached the box, deliver
    Reset everything if successful deliver
```

Program Flowchart

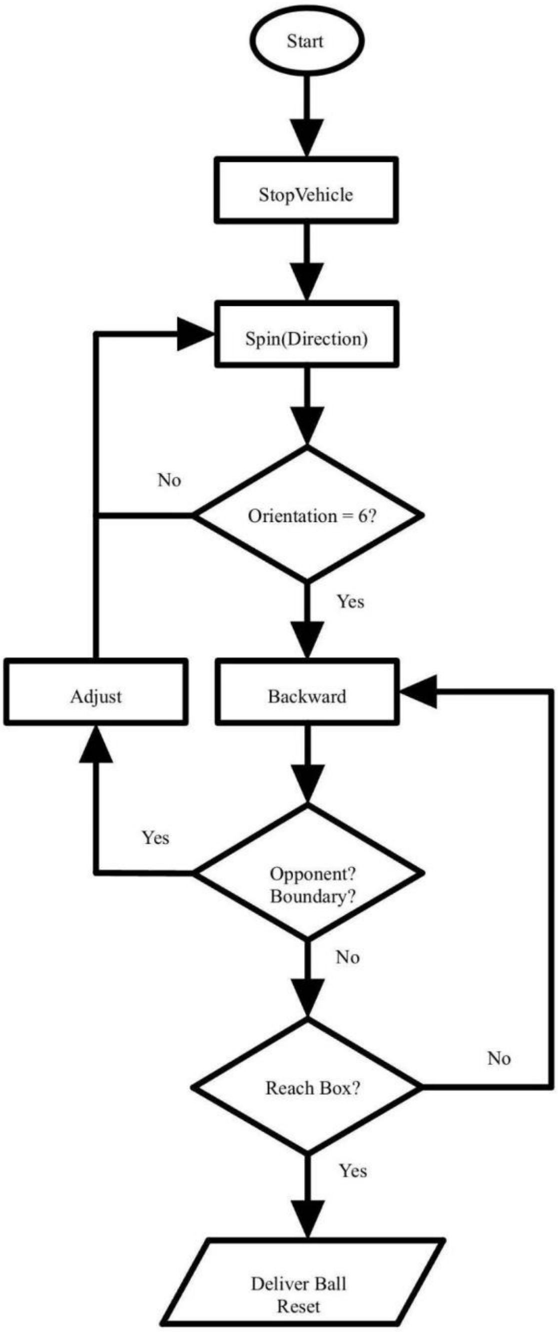


Figure 67: Ball Delivery Program Flowchart

7.6 Boundary Avoidance Strategy

Visualization of Boundary Avoidance Strategy

void isDetect( )

Recall our boundary detection system, we have 4 line tracking sensor that insulate a collision box for boundary. Figure 68 is a visualization of boundary avoidance strategy.

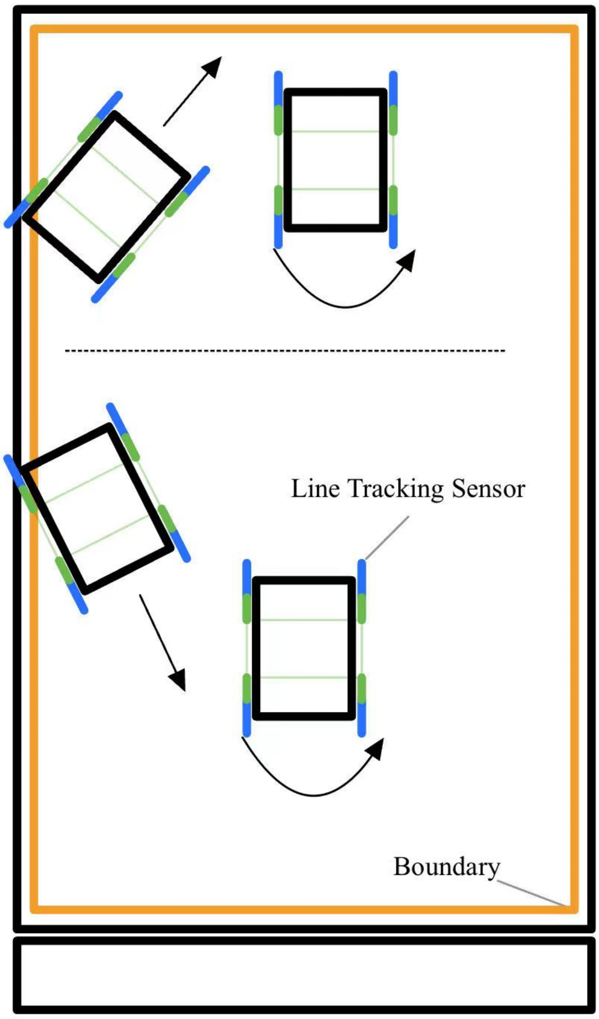


Figure 68: Visualization of Boundary Avoidance

Boundary avoidance system is running parallelly with main program. In this case, regardless of what task is main program performing, as long as our vehicle detected boundary, it will stop the task and execute boundary avoidance.

Figure 68 only shows 2 cases on the left, those cases on the left is symmetric about it. In such a case we can just add a minus sign to the parameter on left.

When back left line tracking sensor detects boundary line, our vehicle will first move forward a bit, and then correct its orientation based on the compass output. Similarly, when front left sensor detects boundary, our vehicle will first move backward, and then adjust its orientation.

Do note that, our searching strategy's direction is depending on boundary detection. To illustrate this, we need to classify the sensors into 2 groups, left/right group and front/back group. Initially we set the searching direction to be search forward, once front/back line tracking sensor detects boundary, the searching direction will change to backward/forward, respectively. Similar for right/left group, initially we set searching direction to spin to the right, once right/left group detects boundary, the searching direction will change to left/right respectively. And all these are controlled by 2 parameter, namely rotationDirection and moveDirection.

Limitations

Our boundary correction system heavily depends on digital compass, however, digital compass outputs are inaccurate sometimes. Therefore, our vehicle keep hitting same boundary same place several times before it can successfully correct it self.

Program Pseudocode

```
void isDetect( )
        Read line tracking sensors
        Switch cases:
        Case back left:
forward, correct.
set rotationDirection = -rotationDirection
set moveDirection = -moveDirection
Case back right:
forward, correct.
set rotationDirection = -rotationDirection
set moveDirection = -moveDirection
Case front left:
backward, correct.
set rotationDirection = -rotationDirection
set moveDirection = -moveDirection
Case front right:
backward, correct.
set rotationDirection = -rotationDirection
set moveDirection = -moveDirection
```

Program Flowchart

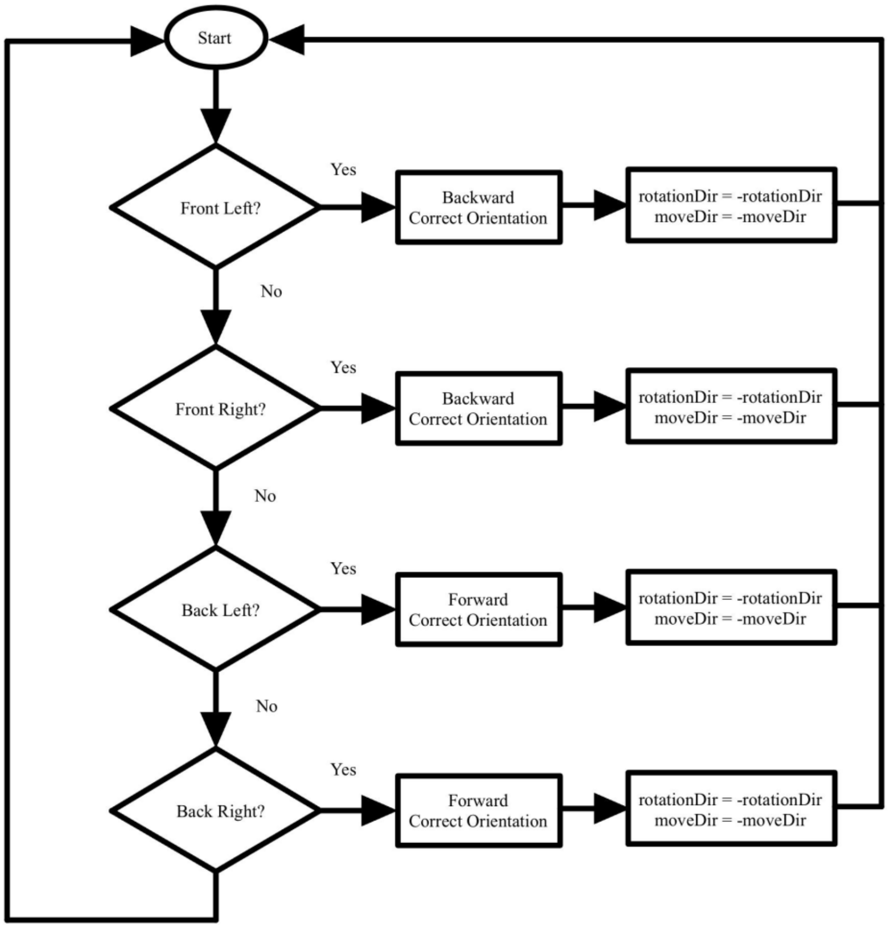
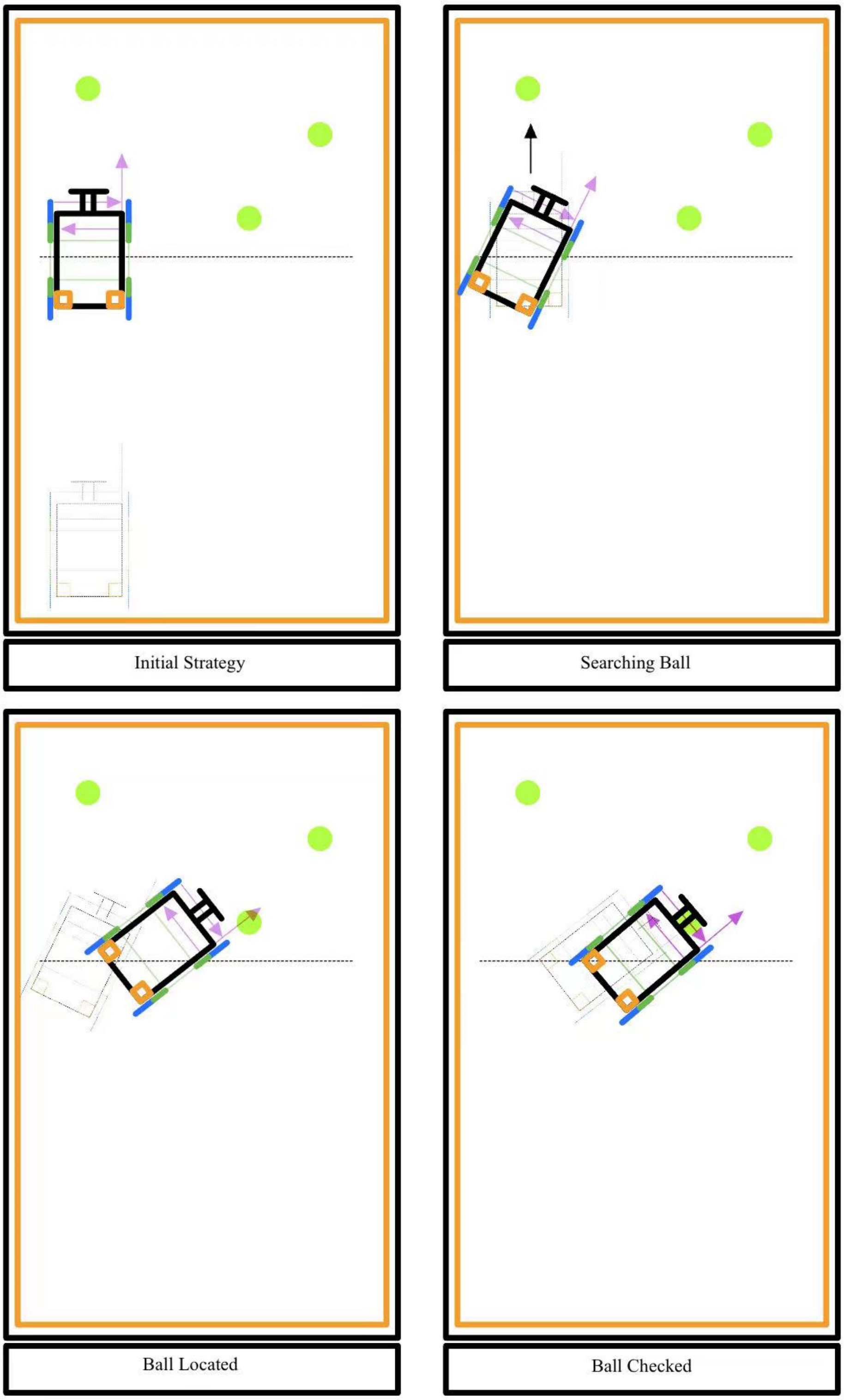
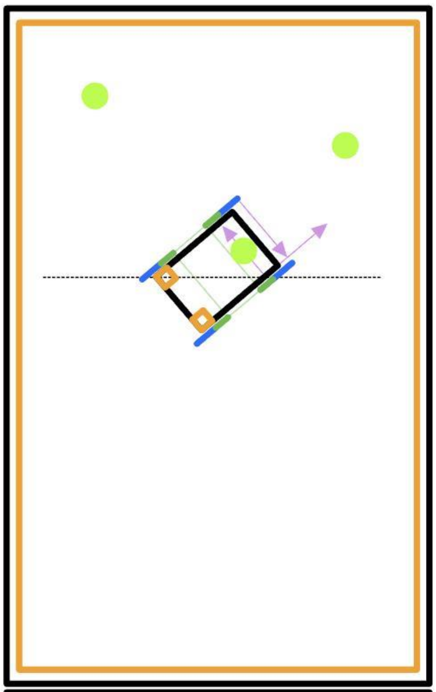


Figure 69: Boundary Avoidance Program Flowchart

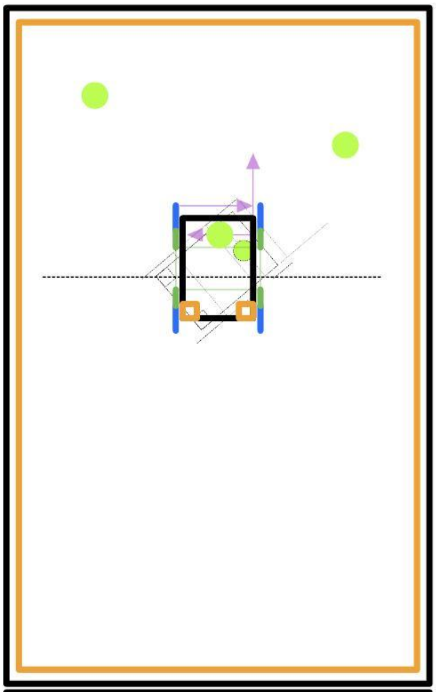
7.7 Main Program Strategy

Main program strategy visualization

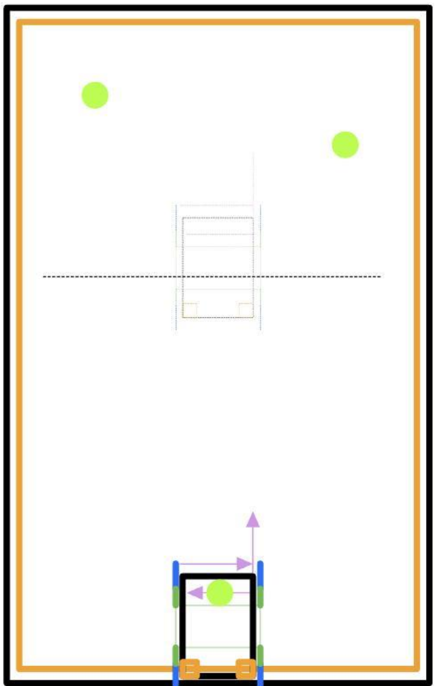




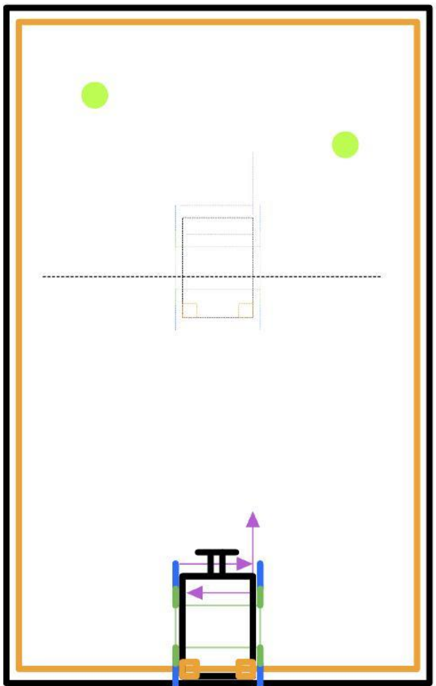
Ball Captured



Correct Orientation



Back to Box



Drop the Ball

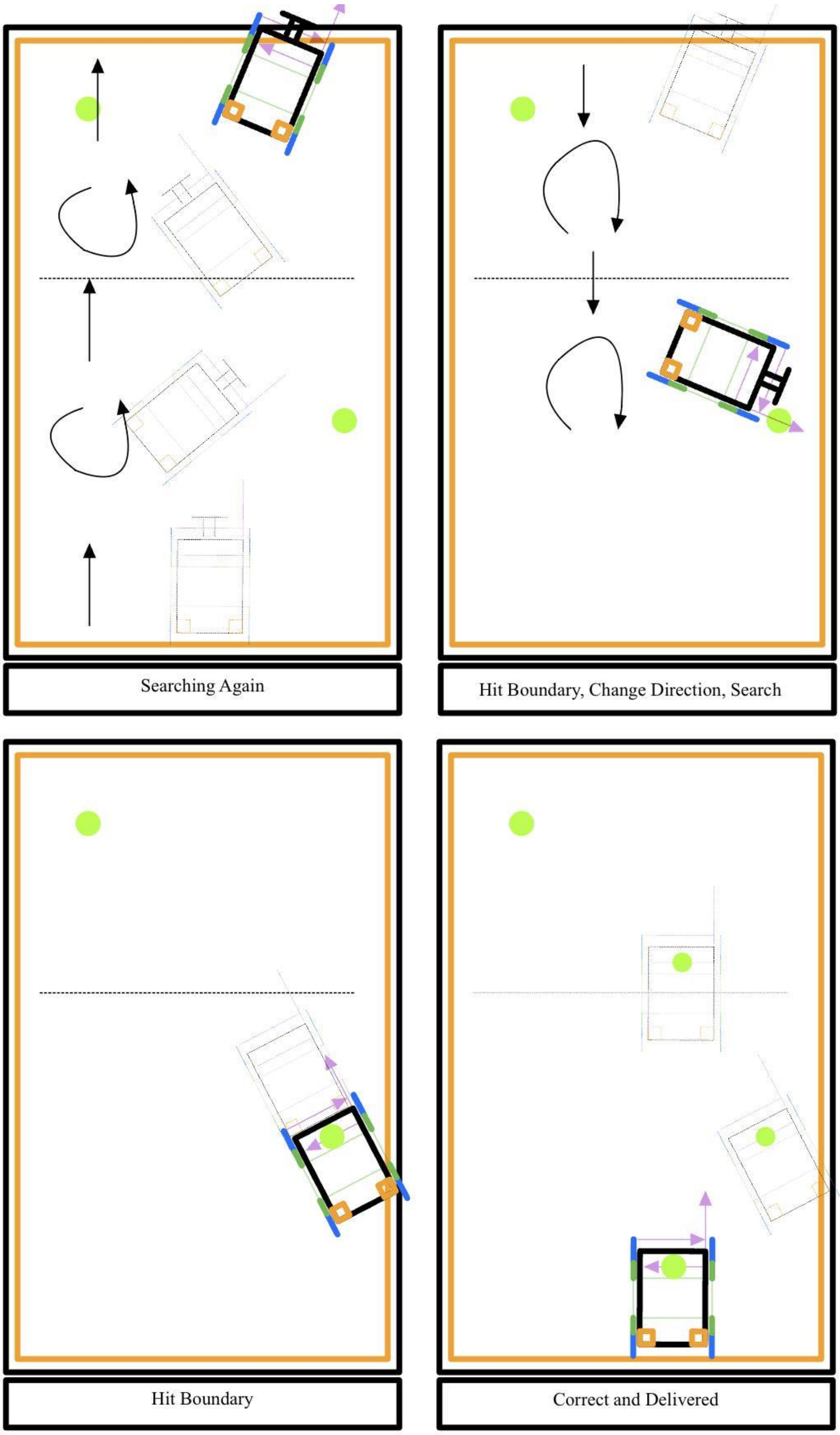


Figure 70: Main Program Logic Visualization

Main Program Flowchart

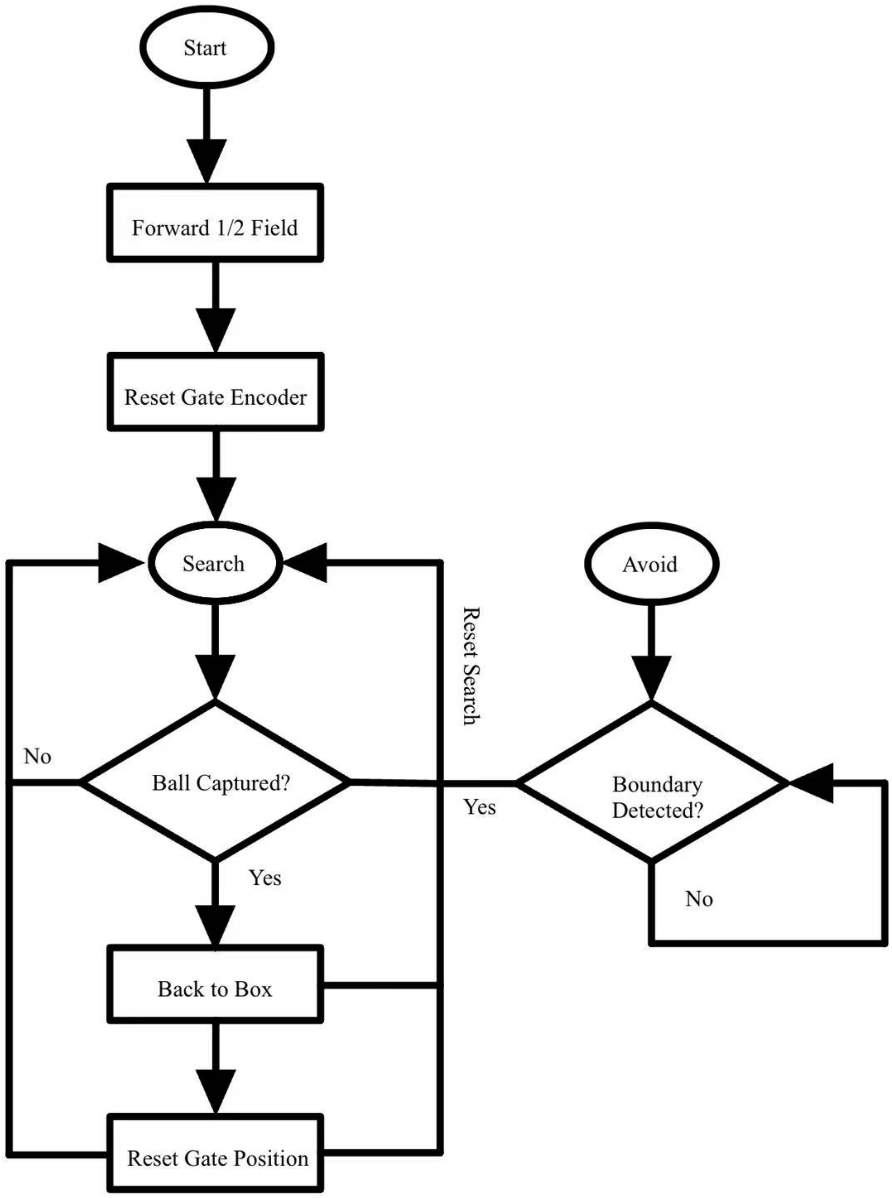


Figure 71: Main Program Strategy Flowchart

Our main program uses multithread programming. After initial strategy, main program will start 2 tasks at the same time, namely search and avoid. These 2 tasks run parallelly and independently. However, task Avoid has higher priority, therefore whenever task Avoid is activated, it will immediately halt task Search to perform boundary & opponent avoidance.

8. Performance in the Competition

8.1 Match and Results

Three teams were involved in the competition. We selected our team name to be FFA and the remaining two teams were called Toyota Avanza and Top Spin. Each vehicle was to compete with the other two opponents and the team with the most number of wins won the competition. In the event that any two teams had the same number of wins, the vehicle that collected the most number of balls was declared the winner.

Our vehicle first went against Toyota Avanza. Even though our vehicle had a lead and advantage at first by securing one ball swiftly, we ran into a hiccup when we missed the second ball delivery. This happened because the gate was not positioned vertically at 90 degrees, it was more inwards. This resulted in lower acceleration of the ball when the delivery was attempted. Fortunately, the opponent team was also stuck at the boundary of the arena and we managed to block them. This results in a 1-1 draw against Toyota Avanza.

Second round was against Top Spin. We had a lead by delivering one ball at the beginning. However, we got stuck when Top Spin collided into us and pushed us to the boundary of the arena. However, Top Spin did not catch this opportunity and was not able to retrieve balls in the remaining time. This resulted in a 1-0 victory against Top Spin.

As Toyota Avanza had a 1-0 victory against Top Spin as well, there was a tiebreaker round conducted between us and Toyota Avanza. Whichever vehicle retrieved one ball first won the competition.. The match began with a rough start as our vehicle collided into Toyota Avanza and lost control at the boundary. However, Toyota Avanza violated a competition rule by collecting two balls at once, and not releasing at least one of them. This resulted in disqualification, and a rematch was conducted. During the second round, Toyota Avanza was quicker with ball detection and secured the ball before us. Consequently, we placed 2nd in the competition.

8.2 Advantage over other teams

During the competition, we noticed that our vehicle had certain advantages over the other vehicles which resulted in our vehicle outperforming other teams in some aspects.

Firstly, our vehicle was light compared to the others. This resulted in the vehicle being the fastest among the three even when the motor speed was not set to the maximum possible speed. Also, our vehicle used one less motor compared to the other groups. This lowered the design complexity of our vehicle, reducing the work of the programming team.

Overall, the lighter weight and lesser electronic components used means that our vehicle has lesser energy consumption, making it energy efficient. The vehicle is capable of operating for the whole day without running into battery drought.

8.3 Vehicle Limitations

The biggest drawback of our design was its reliance on the digital compass for the searching algorithm. Due to its inaccuracy, the vehicle faced problems in terms of manoeuvrability, particularly for delivery of the ball. Moreover, the compass's consistent offset to the left of the vehicle means that the search algorithm was constrained to always start searching from the left.

9. Team Reflection

9.1 Mechanical Team

Team Member	Reflection
Daniel Low Teck Fatt	I would like to share about the area that we have done well and the improvements that can be made to our current design. First of all, I am grateful to be part of the mechanical team as we worked together effectively and scheduled frequent meetings to complete the vehicle structure, so that the software team had ample time to test their software on the vehicle. Regarding the hardware design, we did very well in designing the best ball capturing mechanism, a fact that is acknowledged by the winner of this competition. We have implemented some design to make our vehicle better: 1. Protected the wheels and gears using the metal frame. 2. The font chassis is designed to have the angled metal bar to direct the ball to our collector gate. 3. Our design is clean and lightweight because we only use 3 motors and most of our parts are 3D-printed. In terms of the area for improvements, damage to the ramp and collector gate could have been catastrophic due to our heavy reliance on 3D printing. Malfunction of the ball collection and delivery mechanisms can jeopardise our participation in the game. Furthermore, we overlooked the dimension of the vehicle in CAD when the collector gate was opened 90 degrees to the front as it exceeded the 30 cm limit by 0.8cm. This cost us not being able to install metal frames to protect our gate from collision due to the limitation in dimension. We reprinted shorter swings to make the gate shorter but this adversely affected our ball capture performance. Although we finished the construction of the vehicle structure early, there were many changes to the mechanical design after discovering the underlying problems during the performance testing. As a result, we were still working on the hardware installation even on the competition day itself. Overall, it was a fruitful experience to build an autonomous robot from scratch. It not only showcases the technical skills that we have learned in the past, but also developed essential soft skills such as teamwork and problem solving. Despite the challenges and setbacks faced during the design and testing phase, the sense of accomplishment in bringing our conceptual idea to life was rewarding.
Lim Jia Jing	In summary, the competition went well. I am proud of our team to be able to deliver the robot and tasks and eventually won 2nd place. I did not expect the mechanical team to work straight from the start till the end. There was a constant change of the design since what we planned or envisioned did not perform as per expected in the real world with external factors from the actual environment. The design can never be perfect as there is always room for improvement. In order to perfect it , I believe we can achieve it but only time limits us. I would say that the 3D printed parts are well designed but we did not expect the printed parts to have flaws such as slight sizing which is a critical blow to us. For eg. Some 3D parts are designed to be fixed onto the robot yet due to the flaw in printing, it moved slightly. Overall, the design of our robot was the simplest yet efficient. Since our chassis is well designed to

	protect the components that we have and it is also light weight as compared to others, our robot is strong, fast and robust. Our collecting and depositing ball mechanism is also superior to others since only one motor is required to be used, reducing dependencies and weight . Our 3D printed parts such as the ramp and collector do the job , but it could be better to ensure consistency in depositing since in the competition there was an occurrence that our ball did not manage to go over the ramp. This is something we can foresee before the competition but we did not foresee it well, it is a learning point for us. The use of 3D parts may not be the most ideal but it truly allows us to have the flexibility to customise and not be limited by parts. Other than that, I felt that we took too long to decide which mechanism to use for our robot. We could have tested it out and changed concurrently and decided after testing everything. It will be much faster and decisions made can be based on testing instead of just theoretical reason. With all the modification and improvement made along the way our ball collecting mechanism improved drastically and it works much better than initially. This project really pushes us to the boundary in terms of both technical skills and soft-spoken skills. I have learnt a lot through this project and with the help of my teammates the process was much more enduring. Seeing our product in action is truly a rewarding experience regardless of the competition outcome. It is the process that makes this bitter experience sweet. It is a truly fruitful experience.
Song Ke Yan	As a person more inclined towards the software part, I volunteered to be a part of the mechanical team to gain more skills in mechanical design and contribute to this project. I believe that with the experience in both areas, the design process can be done thoroughly by considering both hardware and software specifications. During the design process, we are able to avoid and foresee a few grave challenges when using SolidWorks to assemble our vehicle. For example, the height of the vehicle was firstly not enough to allow the ball to pass through. We tested a lot with gears and wheels to find the best configuration that suits our vehicle. Initially, we used middle gear drive, but realised it’s not turnable using our hands, but we knew it can be done using motors by supplying larger torque. However, rear wheel drive provides easy turning, almost effortlessly. We continued with this design because if torque provided by hand is enough, then motor torque will make the wheel turn even faster. The result of the competition proved us right when we saw that our vehicle has the highest speed. Overall, there are a few improvements that could be done. I realised that our design is highly dependent on 3D printed parts. This frustration arose when we realised the ramp could have been lower and shorter to ensure higher chance of ball delivery during testing, which eventually caused a backfire during the actual competition. 3D printing could be helpful in some ways, but the main features of our vehicle which are ball capture and ball delivery should not depend too much on 3D printed parts. More flexibility and error tolerance should be allowed. I did not realise until testing that with the motor power supply, it could barely push our ball to the top of the ramp. Foreseeing this kind of situation is what I had not done properly. Also, I should have communicated earlier with the software team to do testing on the motor capabilities. Although I thought that I had prepared a lot for the unexpected circumstances, the reality is that there are more that we never foresee. We had neglected the placement of sensors in the CAD, causing us to have to improvise. Luckily, there are rooms in our

	vehicle for such arrangements. During the testing, more and more modifications are needed on the placement of sensors, such as changing it from horizontal placement to vertical placement, adjusting the height and positions of the line tracking module, and changing the sensor for ball detection from a limit switch to a sharp distance sensor. However, it amazes me how our ball detection mechanism improved overtime with these multiple small changes. We also made a huge change to our base, which worked well in our favour. I learnt that the first plan should only be used as a quick starting guide, persistent works, testing and improvements are what pushes our vehicle performance greater. It is very satisfying to witness our vehicle from zero to a fully autonomous robot.

9.2 Electrical and Programming Team

Team Member	**Reflection**
Liu Qiyuan	As the leader of the VEX RobotC Tennis Ball Fetcher project, named FFA (Far Far Ahead), I encountered and overcame various challenges throughout the project's life cycle. Coordinating the team proved to be a significant difficulty due to conflicting schedules and the chaotic nature of meetings. Drawing on the lessons from the MA4012 course, I applied effective design principles to streamline our processes and enhance overall efficiency during team meetings. In the initial stages, brainstorming for a design concept required collaborative effort. I implemented a strategy where each team member generated at least three ideas, and we used a morphological chart to organise and refine these concepts, eventually arriving at a finalised project concept and prototype. Once the ideas were solidified, the project moved into the implementation phase. The team was divided into hardware and software, with my focus on the software aspect. Despite my role, I closely monitored the hardware development, actively engaging in communication with the hardware team to align their design choices with software requirements. Upon the completion of the hardware phase, I transitioned to programming. Initially utilising Arduino due to my familiarity, I set up the sensor circuits and commenced programming on the Arduino platform. However, during logic testing, I encountered challenges with my initial approach of using elapsed time to distinguish between tennis balls and opponents. Realising the impracticality of this method in dynamic game scenarios, I pivoted to a more feasible approach using height differences, achieving satisfactory results during Arduino testing. The hardware and software integration phase highlighted the limitations of Arduino compared to VEX. To address this, I embarked on learning ROBOTC programming, a decision that greatly benefitted the project. Programming in ROBOTC presented its own set of challenges, primarily centred around sensor inaccuracies. Calibrating sensors was a necessity, but some, like the digital compass, proved difficult to correct through hardware adjustments. Acknowledging the limitations, I devised software solutions in the form of a filter and compensator to mitigate the impact of digital compass errors, ultimately achieving

	success. In the final stages of integrating the main logic, I ventured into multi-thread programming to enable parallel execution of searching and avoidance functions. While this demanded significant time investment, the decision proved valuable in enhancing the overall efficiency and functionality of the system. In conclusion, leading the VEX RobotC Tennis Ball Fetcher project was a rewarding experience that honed my organisational, design, and programming skills. Overcoming challenges, adapting to unforeseen circumstances, and implementing innovative solutions contributed to the success of Team FFA's project.
Cao Xubin	As a software team member in this project competition, I experienced the highs and lows of ideation, programming and testing out a functioning autonomous vehicle. Initially I was feeling uncertain about the challenges ahead but was grateful for my team's encouragement and trust in my opinions and contributions, hence I wish to firstly express my appreciation to them for our collective success. Reflecting on the project's preliminaries, I encountered unfamiliar concepts in systematic idea generation. While initially struggling to generate ideas, I learned the key was to balance critical thinking with constructive contributions, to criticise our suggestions and not get held on by a single idea. Our team's diverse perspectives proved essential in refining and selecting optimal ideas. In the project's first half, the focus was on the main collection and drive system and initially using the Arduino for the software sensor portion. However, we faced critical issues and eventually switched to the VEX microcontroller. Adapting to RobotC programming for VEX posed a major challenge, but with online tutorials and the software team's support, I gained proficiency in this area. This experience reinforced the value of persistent effort in mastering new skills where one can quickly turn something unfamiliar and familiar with constant efforts. Calibrating sensors also emerged as a significant obstacle, while being pressured by the competition deadlines. Testing and refining navigation, detection, and collection logic proved demanding and frustrating as it required creative problem-solving and out of the box thinking. However, with each logic and functions that tested successfully came affirmation and confidence which kept our software team pressing onwards. Unexpectedly, hardware adjustments to overcome software struggles became crucial, highlighting the importance of collaboration between software and the mechanical team. While this reflection only depicts only the most major challenges I faced, the rich and intricate learning experiences throughout the whole process would remain vivid and impactful to me. The enhancing theoretical knowledge, honing of my problem solving skills and the collaborative efforts within my team unveiled my strengths and limitations. Being able to get recognised and overcoming personal limits makes this journey truly rewarding.

Selvam Vinothini Vibhusha	This project has been one of the most interesting and hands-on projects that I have come across in NTU. I am really proud of what we have achieved as a team as we did our best to ensure our robot is prepared for any scenario we could think of during testing. Although we placed overall second due to situations beyond our control, it was rewarding to see our robot with the most efficient collector mechanism and quick search strategy. We made the right decision to combine the electrical and software team into a single group as this allowed us the freedom to experiment with the electronic placements to compliment our code. This proved especially useful when trying to figure out where to keep the distance sensors so that we get useful readings without compromising the searchable area too much. The conceptual design phase was the most critical phase. It showed me how easy it is to get lost in the brainstorming process to find the "perfect" design, eventually losing sight of the big picture which is to essentially to produce a working prototype that is capable of collecting the ball. Because of this, I became more aware in prioritising what are important tasks for the vehicle and what are added features that make it superior to the other teams. By keeping this distinction in mind, it was much easier to start with the programming. I worked on the motor functions and digital compass calibration along with Xiaoni. Basic programs were first written to test that the components can work. After this, basic functions to be used for the search algorithm were developed and tested before combining everything altogether. This iterative process allowed progressive development that we could keep track of and understand what was happening. One of the biggest bottlenecks we faced was that the RobotC IDE was not compatible with our laptops. This slowed down our progress as we could not resolve the issue despite consulting with the lab technician and even changing the VEX microcontroller we were using. We resorted to using the PCs in the lab as it worked fine there. After working separately, we collaborated with the sensor team (Qiyuan and Xubin) to brainstorm ideas for the search algorithm. Testing was the most tedious process. While testing, we spent a significant amount of time deciding how the vehicle will detect the arena boundaries. When we thought we resolved the issue of the vehicle crossing the boundary, we detected a clash between the search algorithm and boundary detection algorithm for control over the motor. Sufficient to say, testing was the most important stage as it helped us to see what we were missing that could make our vehicle better. Overall, these are some of the key moments that standout to me. I am really grateful to be able to learn how to design systematically like an engineer through this project.

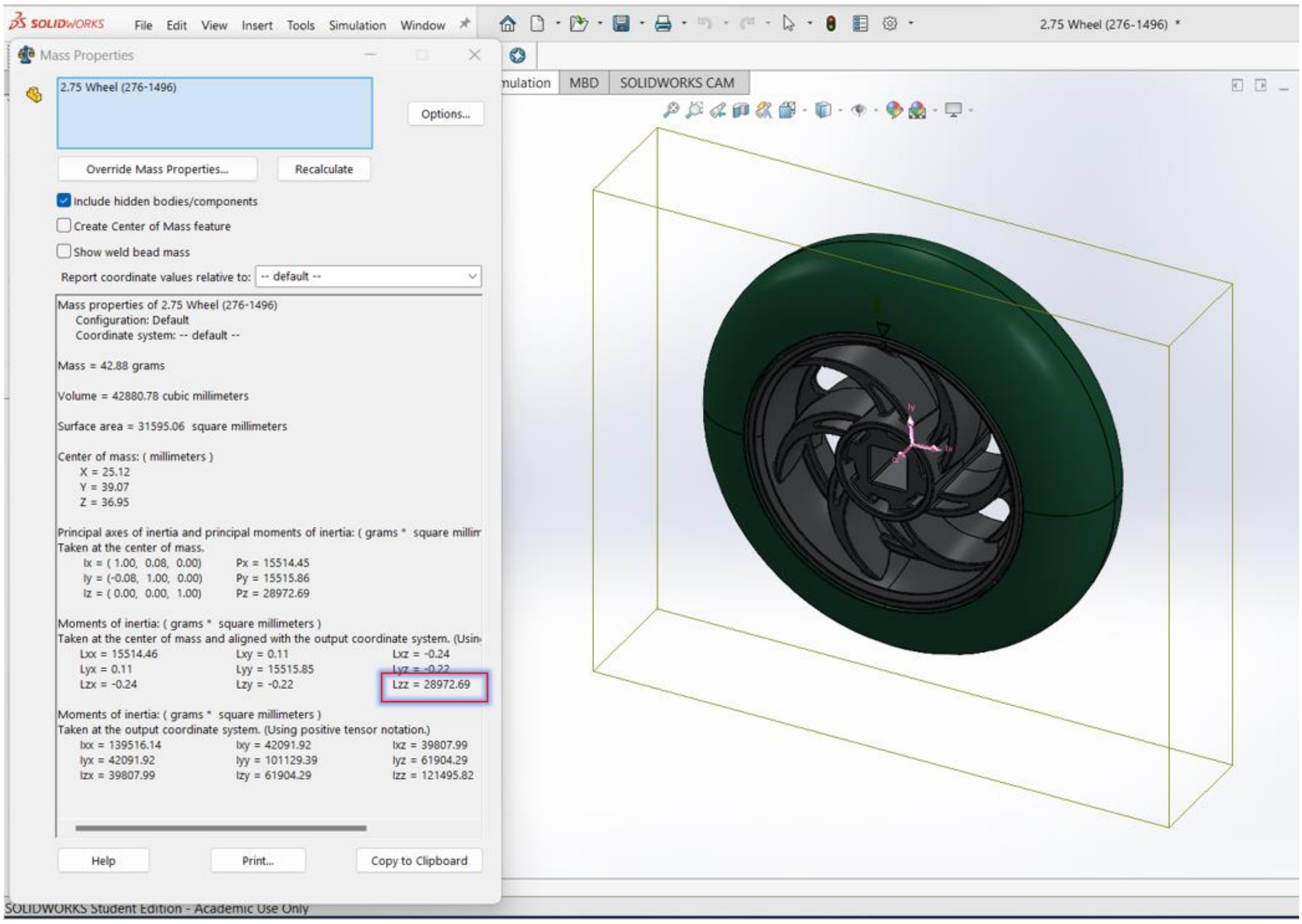
Wang Xiaoni	Reflecting on the development of robots compared to pure software development, I found it to be more challenging. Many times, errors occurred without detailed run logs, making debugging more difficult. And the lack of open-source resources for reference added to the complexity to the process. Developing the motion processes for robots on the VEX platform proved to be more fun than I had anticipated. VEX, as an IDE, is not very user-friendly, with various modes for controlling robots, and occasional firewall configuration issues hindering startup. Throughout the development, limitations in sensor sensitivity, such as the occasional fuzziness in compass direction values, collector posed challenges. For instance, even with the initial angle set for servos, gravitational issues sometimes prevented them from returning to their original position. The most challenging aspect of the entire project, in my opinion, lies in addressing edge cases—situations that are hard to predict without practical experimentation. An example includes the tennis ball getting stuck at the initial point of receiving, preventing the switch at the tail from activating. Despite extensive debugging, unforeseen special situations were encountered during competitions, such as the robot getting stuck at the edges and moving back and forth. Brainstorming within the team emerged as the most effective method for overcoming these challenges. At times, innovative ideas emerged that deviated completely from existing solutions. Maintaining simplicity and clarity in logic always impressed me. This project represents the most formal robotics development experience in my four years of university. The rules like opponents added complexity to the development process. Overall, it was a great learning opportunity that allowed me to become familiar with C language development and understand the operational logic of robots.

10. Appendix

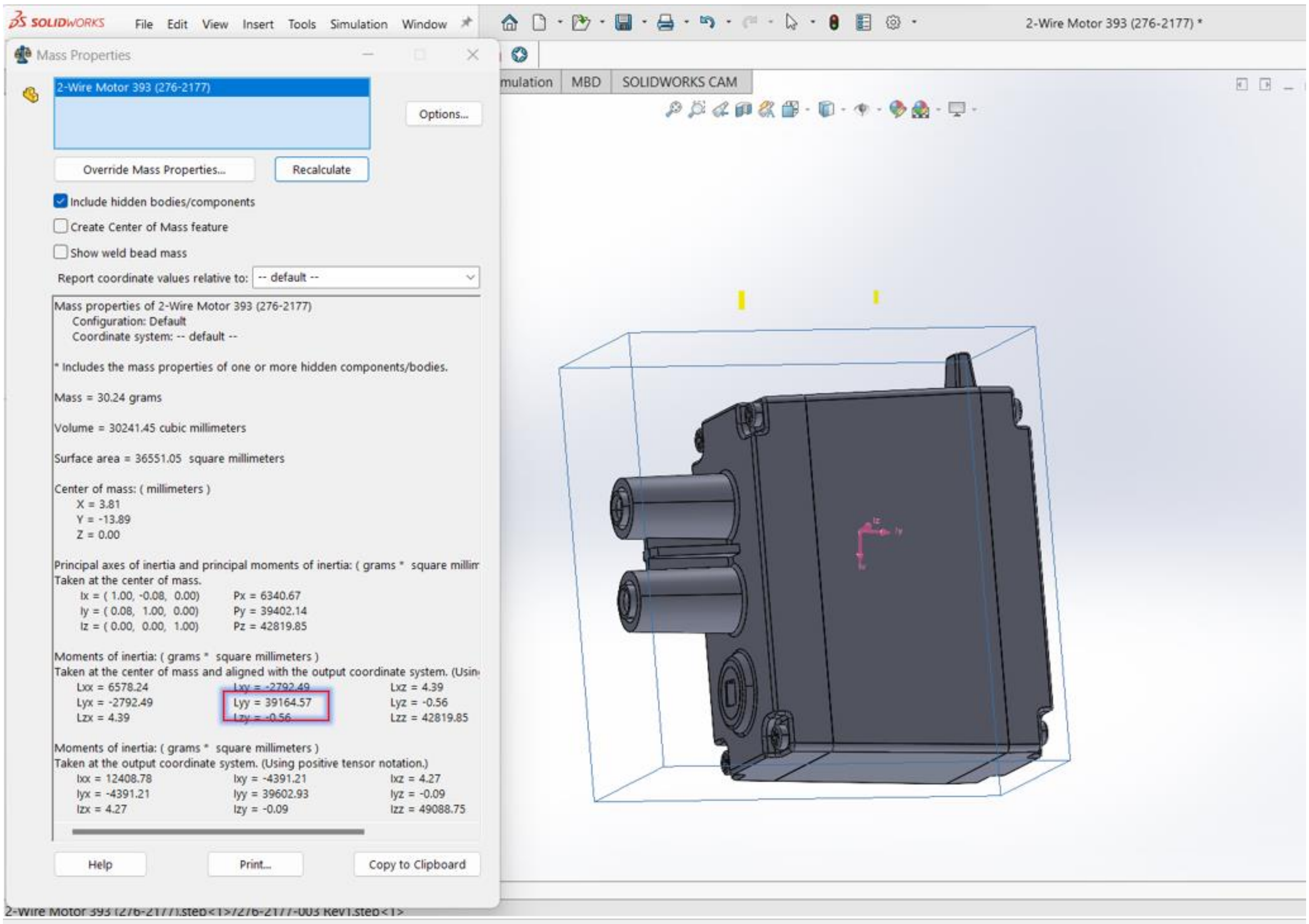
Appendix A: Moment of Inertia

Moment of Inertia, J of Components

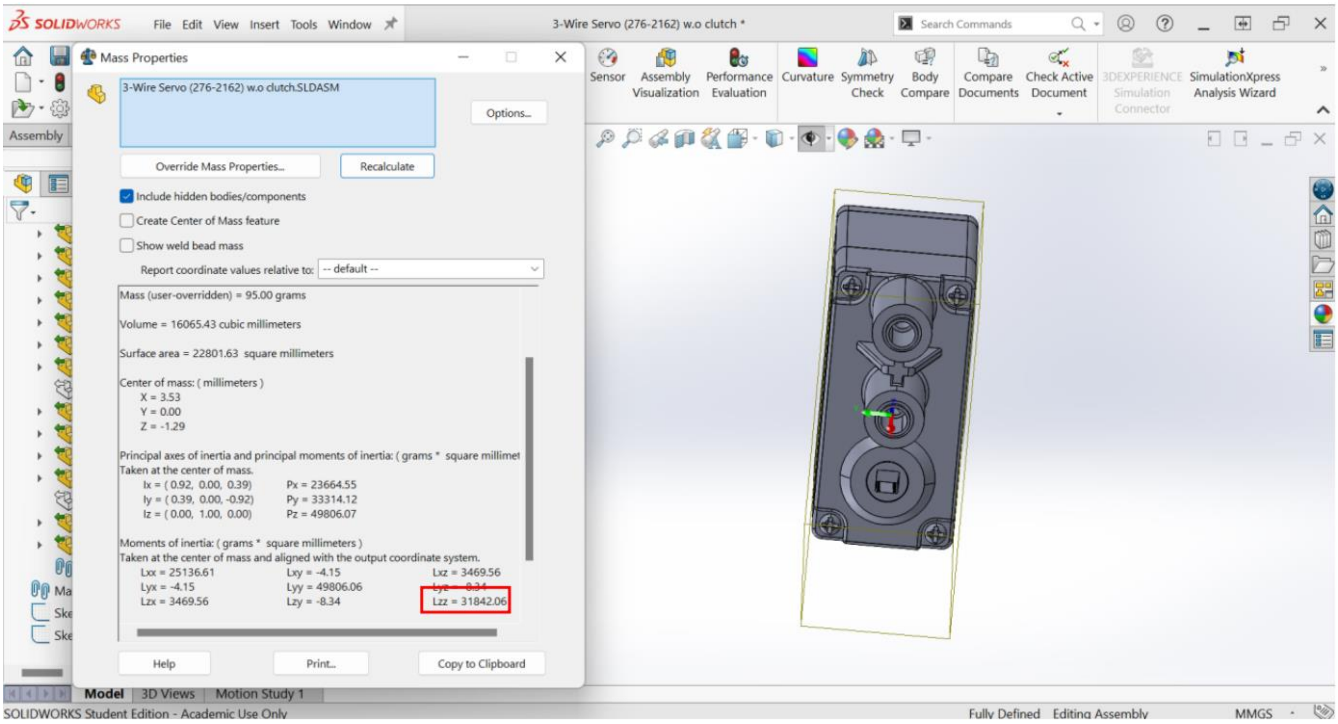
Wheels:



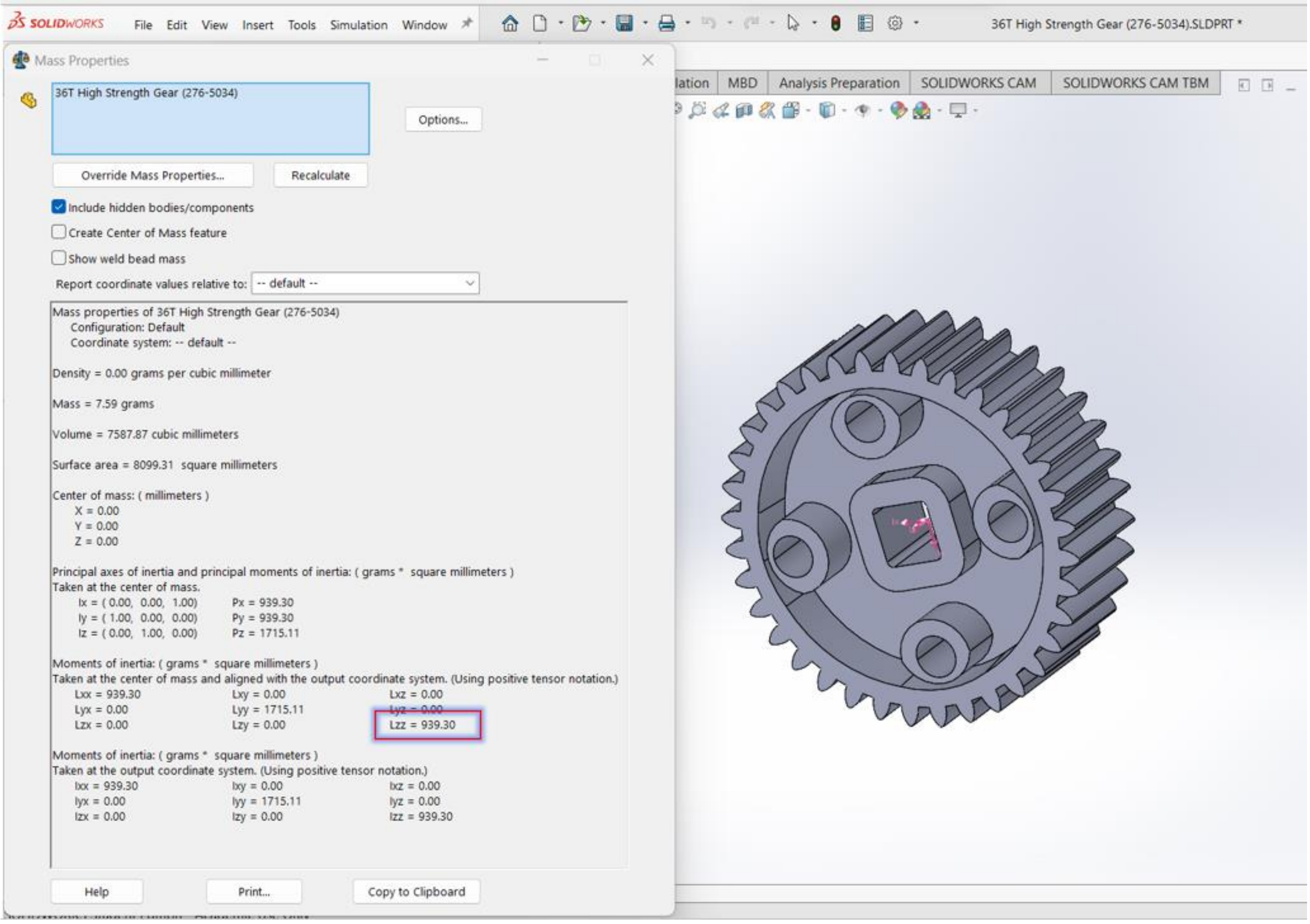
2- Wire Motor



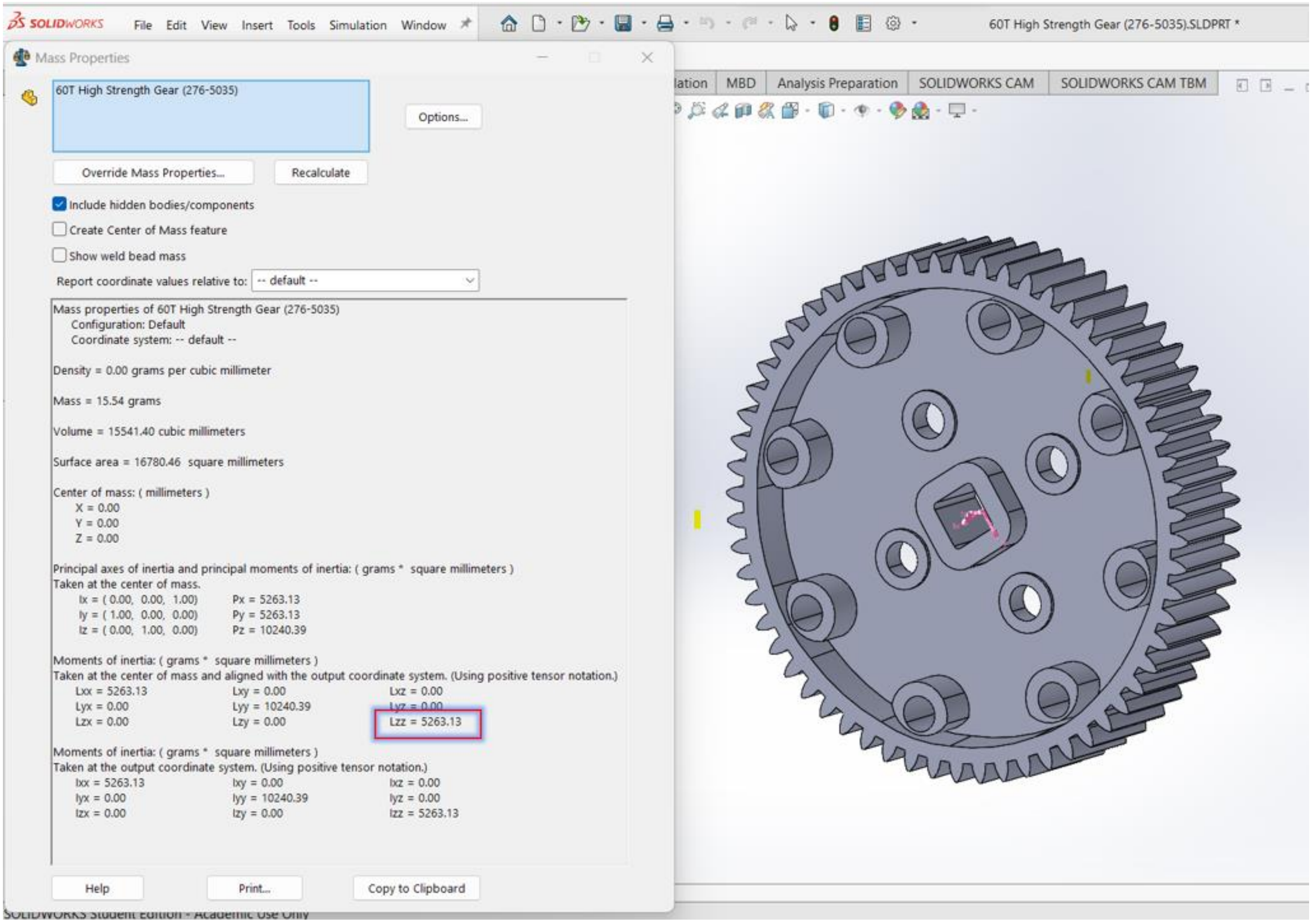
3-Wire Motor



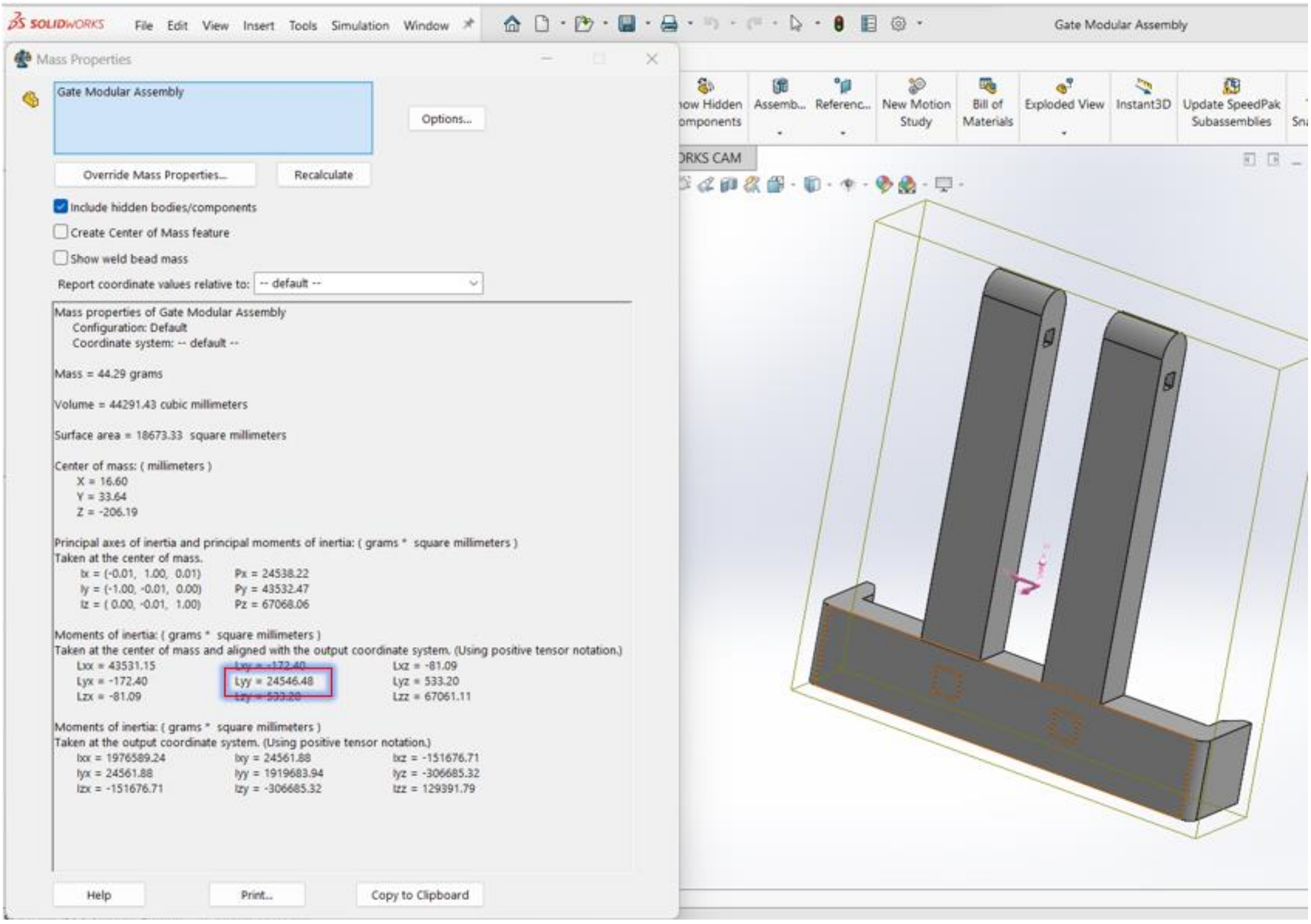
36T High Strength Gear



60T High Strength Gear



Collector Gate



Appendix B: Source Code

```
#pragma config(Sensor, in1,    IR1,           sensorAnalog)
#pragma config(Sensor, in2,    IR2,           sensorAnalog)
#pragma config(Sensor, in3,    IR3,           sensorAnalog)
#pragma config(Sensor, in4,    IR4,           sensorAnalog)
#pragma config(Sensor, dgtl1,  gateEncoder,   sensorQuadEncoder)
#pragma config(Sensor, dgtl3,  LineTracking1, sensorDigitalIn)
#pragma config(Sensor, dgtl4,  LineTracking2, sensorDigitalIn)
#pragma config(Sensor, dgtl5,  LineTracking3, sensorDigitalIn)
#pragma config(Sensor, dgtl6,  LineTracking4, sensorDigitalIn)
#pragma config(Sensor, dgtl7,  LimitSwitchL,  sensorTouch)
#pragma config(Sensor, dgtl8,  LimitSwitchR,  sensorTouch)
#pragma config(Sensor, dgtl9,  Compass1,      sensorDigitalIn)
#pragma config(Sensor, dgtl10, Compass2,      sensorDigitalIn)
#pragma config(Sensor, dgtl11, Compass3,      sensorDigitalIn)
#pragma config(Sensor, dgtl12, Compass4,      sensorDigitalIn)
#pragma config(Motor,  port2,          vexGate,       tmotorServoContinuousRotation, openLoop)
#pragma config(Motor,  port3,          vexBaseR,      tmotorVex393_MC29, openLoop, reversed)
#pragma config(Motor,  port4,          vexBaseL,      tmotorVex393_MC29, openLoop)
//*!!Code automatically generated by 'ROBOTC' configuration wizard               !!*//

//___________________________________________________________________________________

#include "utilities/searchUtilities.c"
#include "utilities/motorUtilities.c"
#include "utilities/transportUtilities.c"
#include "searchBall.c"
#include "moveBall.c"
#include "avoidBoundary.c"

task boundary() {
  while (true) {
    isDetect();
  }
}

//___________________________________________________________________________________

task main(){

  // preparation command run here once
  moveForward(120, 3500);
  SensorValue(gateEncoder) = 0;

  // main program stream run here repeatedly
  while (true){
    startTask(boundary);
    if (!LineTrackTriggered) {
      stopRobot();
      searchWhateverInfront_version2();
      LineTrackTriggered = false;
    }
    if (ballCaptured == true) {
      stopTask(boundary);
      backToBox();
    }
    if (ballCaptured == false) {
      gateControl(0);
    }
  }
}

//___________________________________________________________________________________
```

```
bool ballDetected = false;
bool ballCaptured = false;
int searchDirection = 50;

void searchWhateverInfront_version2() {
  rotate(rotSpeed);
  clearTimer(T1);
  while (time1[T1] < 5000) {
    if (isObjectInfront(IR3, 500)) {
      if (!(isObjectInfront(IR1, 500))) {
        stopRobot();
        if (rotSpeed > 0) {
          rotateWithTime(-rotSpeed, 350);
        }
        else if (rotSpeed < 0) {
          rotateWithTime(rotSpeed, 350);
        }
        forward(30);
        clearTimer(T1);
        while (time1[T1] < 3000) {
          if (isBall(IR2)) {
            gateControl(-85);
            break;
          }
        }
        if (isCapture(IR4)) {
          ballCaptured = true;
          stopRobot();
        }
        break;
      }
      else {
        backWithTime(25, 500);
      }
    }
  }
  int orn = readCompass();
  while ((!ballCaptured) && orn != 6) {
    rotate(rotSpeed);
    orn = readCompass();
  }
  if (!ballCaptured) {
    moveForward(searchDirection, 800);
  }
```

```
void moveBackward(int speed) {
    motor[vexBaseR] = speed;
    motor[vexBaseL] = speed;
}

void backWithTime(int speed, int msec) {
    motor[vexBaseR] = speed;
    motor[vexBaseL] = speed;
    wait1Msec(msec);
    motor[vexBaseR] = 0;
    motor[vexBaseL] = 0;
}

void forward(int speed) {
    motor[vexBaseR] = -speed;
    motor[vexBaseL] = -speed;
}

void moveForward(int speed, int msec) {
    motor[vexBaseR] = speed;
    motor[vexBaseL] = speed;
    wait1Msec(msec);
    motor[vexBaseR] = 0;
    motor[vexBaseL] = 0;
}

int rotSpeed = 40;
void rotate(int speed) {
    motor[vexBaseR] = speed;
    motor[vexBaseL] = -speed;
}

void rotateWithTime(int speed, int msec) {
    motor[vexBaseR] = speed;
    motor[vexBaseL] = -speed;
    wait1Msec(msec);
    motor[vexBaseR] = 0;
    motor[vexBaseL] = 0;
}

void stopRobot() {
    motor[vexBaseR] = 0;
    motor[vexBaseL] = 0;
}

int indexing(tSensors sensorPin) {
  // Function to index sensor pins
  int sensorIndex;
  if (sensorPin == IR1) sensorIndex = 0;
  else if (sensorPin == IR2) sensorIndex = 1;
  else if (sensorPin == IR3) sensorIndex = 2;
  else if (sensorPin == IR4) sensorIndex = 3;
  return sensorIndex;
}

//______________________________________________________________________________

int lastValue[4] = {0,0,0,0};
bool isObjectInfront(int sensorPin, int Threshold) {
  // Function to check if an object is in front
  int sensorIndex = indexing(sensorPin);
  int rawValue = SensorValue[sensorPin];
  bool exist = (lastValue[sensorIndex] >= Threshold && rawValue >= Threshold);
  lastValue[sensorIndex] = rawValue;
  return exist;
}

//______________________________________________________________________________
```

```
bool isCapture(int sensorPin) {
  int rawValue = SensorValue[sensorPin];
  if (rawValue > 1000) {
    return true;
  }
  else {
    return false;
  }
}

//___________________________________________________________________________

bool isBall(int sensorPin) {
  int rawValue = SensorValue[sensorPin];
  if (rawValue > 2000) {
    return true;
  }
  else {
    return false;
  }
}

//___________________________________________________________________________

bool isReturn(int LS_back_1, int LS_back_2) {
  bool LSB1 = SensorValue[LS_back_1] == 1;
  bool LSB2 = SensorValue[LS_back_2] == 1;
  if (LSB1 && LSB2) {
    return true;
  }
  else {
    return false;
  }
}

//___________________________________________________________________________
```

```
int compass_status;
int readCompass()
{
    int pin1 = SensorValue(Compass1);
    int pin2 = SensorValue(Compass2);
    int pin3 = SensorValue(Compass3);
    int pin4 = SensorValue(Compass4);
    int combination = pin1 * 1000 + pin2 * 100 + pin3 * 10 + pin4;
    switch (combination)
    {
    case 1110: //NORTH
        compass_status = 0;
        break;
    case 1100: //NORTH_EAST
        compass_status = 1;
        break;
    case 1101: //EAST
        compass_status = 2;
        break;
    case 1001: //SOUTH_EAST
        compass_status = 3;
        break;
    case 1011: //SOUTH
        compass_status = 4;
        break;
    case 0011: //SOUTH_WEST
        compass_status = 5;
        break;
    case 0111: //WEST
        compass_status = 6;
        break;
    case 0110: //NORTH_WEST
        compass_status = 7;
        break;
    default: //INVALID_COMBINATION
        compass_status = 8;
        break;
    }
    return compass_status;
}
```

```
int orn = readCompass();
void adjustDirection() {
    while ((!ballCaptured) && orn != 6) {
      rotate(rotSpeed);
      orn = readCompass();
  }
}

bool LineTrackTriggered = false;

void isDetect() {
  int BL = SensorValue(LineTracking1);
  int FL = SensorValue(LineTracking2);
  int BR = SensorValue(LineTracking3);
  int FR = SensorValue(LineTracking4);
  if (BL == 0) {
    LineTrackTriggered = true;
    stopRobot();
    moveForward(30, 800);
    adjustDirection();

    rotSpeed = -40;
    searchDirection = 50;
  }
  else if (FL == 0) {
    LineTrackTriggered = true;
    stopRobot();
    backWithTime(30, 800);
    adjustDirection();

    rotSpeed = -40;
    searchDirection = -50;
  }
  else if (BR == 0) {
    LineTrackTriggered = true;
    stopRobot();
    moveForward(50, 800);
    adjustDirection();

    rotSpeed = 40;
    searchDirection = 50;
  }
  else if (FR == 0) {
    LineTrackTriggered = true;
    stopRobot();
```

```
bool adjustment = true;
void backToBox(){
  stopRobot();
  while (ballCaptured == true) {
    rotate(rotSpeed);
    int orn = readCompass();
    while (orn == 6) {
      moveBackward(50);
      int deviation = SensorValue(LineTracking1);
      if (deviation == 0) {
        moveForward(50, 1000);
        rotateWithTime(30, 500);
      }
      if (isReturn(LimitSwitchL, LimitSwitchR)) {
        stopRobot();
        gatePush();
        ballCaptured = false;
        ballDetected = false;
        adjustment = true;
        searchDirection = 50;
        moveForward(120, 3500);
        break;
      }
    }
  }
}
```

GP2Y0A41SK0F

Distance Measuring Sensor Unit
Measuring distance : 4 to 30 cm
Analog output type

■Description

GP2Y0A41SK0F is a distance measuring sensor unit, composed of an integrated combination of PSD (position sensitive detector) , IR-LED (infrared emitting diode) and signal processing circuit.
The variety of the reflectivity of the object, the environmental temperature and the operating duration are not influenced easily to the distance detection because of adopting the triangulation method.
This device outputs the voltage corresponding to the detection distance. So this sensor can also be used as a proximity sensor.

■Agency approvals/Compliance

1. Compliant with RoHS directive (2011/65/EU)

■Applications

1. Cleaning robot
2. Personal robot
3. Sanitary

■Features

1. Distance measuring sensor is united with PSD, infrared LED and signal processing circuit
2. Short measuring cycle (16.5ms)
3. Distance measuring range : 4 to 30 cm
4. Package size (29.5 × 13.0 × 13.5mm)
5. Analog output type